TRAJECTORY-BASED LAUNCH VEHICLE PERFORMANCE ANALYSIS FOR DESIGN-SPACE EXPLORATION IN CONCEPTUAL DESIGN

A Thesis
Presented to
The Academic Faculty

by

Michael J. Steffens

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Aerospace Engineering

Georgia Institute of Technology
August 2016

TRAJECTORY-BASED LAUNCH VEHICLE PERFORMANCE ANALYSIS FOR DESIGN-SPACE EXPLORATION IN CONCEPTUAL DESIGN

Approved by:

Professor Dimitri Mavris, Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Professor Daniel Schrage
School of Aerospace Engineering
Georgia Institute of Technology

Professor Marcus Holzinger
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Brad St. Germain
Engineering
SpaceWorks Enterprises, Inc.

Mr. Mark Rogers
Marshall Space Flight Center
National Aeronautics and Space Administration

Date Approved: July 19, 2016

To Jesus Christ, my Lord and Savior

ACKNOWLEDGEMENTS

I take a moment now to express my gratitude to the many people who have helped me along my road here. I have had advisors who have guided me and challenged me, friends and family who have supported me and peers who have encouraged me and critiqued my work. I am grateful to all of them.

To my advisor, Dr. Mavris, thank you for making my graduate studies possible by welcoming me into ASDL. Thank you for always challenging me to strive for excellence. Thank you for your patient guidance as I learn. It has been a privilege to be your student.

I am grateful to my committee, Dr. Mavris, Dr. Schrage, Dr. Holzinger, Dr. St. Germain, and Mr. Rogers, for their guidance and feedback through this process. I appreciate all the time and effort they have put in to help me finish this dissertation.

I owe a special thank you to Stephen Edwards, for generously giving his time to meet with me and for helping me work through each problem as it arose. He was a constant source of guidance through my graduate studies. I am grateful for his friendship, wisdom, and encouragement.

I would also like to express my thanks to all my fellow students and colleagues for their input, feedback, encouragement, and constructive criticism through the years. We have spent long hours together preparing for presentations, finishing projects for classes, studying for the qualification exams, and working on our dissertations. I cherish the friendships I have developed during my time here. I will specifically thank Justin Kizer and Ryan Jacobs for diligently meeting early every Tuesday morning during the last year and a half of this process. Those meetings helped keep me motivated and on track.

To my family and friends, thank you for your love and support. Thank you for believing in me and being there for me. To my wife Sarah, thank you for standing by me through this process, for encouraging me and supporting me when the road was tough. I could not have asked for a better partner to travel this road with.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Trajectory optimization is an important part of launch vehicle conceptual design. It provides the link between a proposed vehicle and its performance, which can be used to compare vehicles against each other and against requirements. This is especially important in early phases of design, where a large design space of vehicles may be considered and must be pared down to a few candidate vehicles. Current methods for trajectory optimization, which involve numerical analysis, are computationally expensive, and require trajectory experts in the loop, do not allow for large design space exploration. A simplified performance analysis, like the rocket equation, is much better suited to the types of studies desired in conceptual design, where thousands of vehicles can be considered and compared. Unfortunately, the rocket equation does not take into account trajectory losses and therefore does not provide an accurate measure of performance. The lack of a fast and accurate method to evaluate launch vehicle performance represents a gap in the current capability that will be addressed in this thesis.

The goal of this research is to formulate and implement a performance analysis method in the form of the rocket equation (i.e. closed-form) that takes into account the trajectory losses considered in a numerical trajectory analysis method. Achieving this goal will result in a capability that enables rapid and accurate performance evaluation of launch vehicles. In conceptual design, this can be used in the context of multi-disciplinary optimization, technology trade studies, probabilistic assessments, and Monte Carlo analysis for launch vehicles. In addition, a much larger vehicle design space can be considered in the same amount of time.

This document lays out the motivation and research approach for achieving the

stated goal. A literature review of the current methods employed in launch vehicle trajectory optimization is first conducted. These include requirement of a user in the loop to accurately find the performance of a launch vehicle. A series of statistical methods from the literature are submitted to address these challenges. The launch vehicle trajectory problem is statistically posed as finding the extreme value of a distribution representing the performance values for all the possible trajectories. This view of the problem is shown to be applicable to optimization problems in general, as well as specifically to the trajectory optimization problem.

Throughout the discussion of the problem, a set of research questions is presented to determine how to apply the trajectory optimization methods in conjunction with the statistical methods. These research questions are addressed either from the literature or through experimentation. For this thesis, the experimentation is conducted using a relevant launch vehicle problem. The trajectory of a Delta IV Heavy to low earth orbit is chosen as the problem. The result of these efforts is a method for evaluating the performance of a launch vehicle without requiring user input, where each step in the method is determined based on the answer to the corresponding research question. While the method does provide accurate performance estimates, it does not do so in the time required to meet the research objective. Therefore, the method is employed to generate a surrogate model of the launch vehicle performance data. Two more research questions address how to generate the surrogate model. The answer to these, along with the previous method for launch vehicle performance analysis, leads to an overall methodology, named RAPTOR, that meets the research goal.

The RAPTOR methodology provides a process that can be implemented to consider the desired vehicle design space in conceptual design. In this thesis, the process is implemented for the Delta IV Heavy with an example design space. The resulting surrogate model is able to accurately estimate the performance of a launch vehicle in the design space of interest virtually instantaneously. This method is flexible and

can be applied to any launch vehicle and any design space of interest. In addition, the uncertainty in the performance estimates is quantified, so additional effort can be leveraged to decrease uncertainty. Several example applications are given to demonstrate the usefulness of the RAPTOR methodology.

The performance analysis capability that results from implementing RAPTOR meets the research objective of enabling rapid and accurate launch vehicle performance analysis in conceptual design. This provides a way of estimating the performance for thousands of vehicles in the design space considered where previously only a few were considered. This affords decision makers the ability to weigh options and ask the "what if" questions in a real time setting with accurate estimates of the performance of the launch vehicle considered.

CHAPTER I

INTRODUCTION

Launch vehicles play an important role in our society today. In addition to enabling the exploration of our solar system and beyond, they provide access to space for military and government applications as well as commercial services, such as telecommunications and weather prediction. The design of a launch vehicle is a complex and time consuming task. Many different disciplines must be considered, each playing a critical role in the capability and affordability of the vehicle. This thesis presents a novel approach for addressing one of the key disciplines in launch vehicle design.

1.1 Challenges of Spaceflight

1.1.1 The Physics of Getting to Space

Launch vehicles are designed, built, and operated with the express purpose of transporting payload from Earth's surface to a specified orbit. At the surface of the Earth a payload will be at 0 km in altitude and moving less than 1 km/s due to the rotation of the Earth. A typical low earth orbit (LEO) mission will transfer a payload from Earth's surface to an altitude of 500 km and accelerate it to more than 7 km/s. A simple calculation shows the difference in energy between the states.

$$\begin{aligned} \Delta E = \Delta KE + \Delta PE = \frac{1}{2}m(v_2^2 - v_1^2) + mg(h_2 - h_1) = \\ \frac{1}{2}m(7000^2) + m(9.81)(500,000) \quad \rightarrow \quad \sim 30MJ/kg \end{aligned} \tag{1}$$

For every kg of payload, about 30 MJ of energy needs to be imparted. This is roughly equivalent to the kinetic energy of a loaded 18-wheeler moving 40 $miles/hour$ for every single kg. This estimate does not take into account any energy losses incurred during the vehicle launch.

A measure of launch vehicle performance can be derived starting with Newton's third law by considering a rocket in space. The derivation is simplified by assuming no drag or gravitational forces and by approximating the rocket as a point mass.

$$F = ma = m\frac{dV}{dt} = Thrust = T \rightarrow \frac{dV}{dt} = \frac{T}{m} \tag{2}$$

Rearranging and multiplying by $1 = \frac{dt}{dm}\frac{dm}{dt}$

$$dV = \frac{T}{m}dt = \frac{T}{m}dt\frac{dt}{dm}\frac{dm}{dt} = T\frac{dt}{dm}\frac{dm}{m} = -\frac{T}{\dot{m}}\frac{dm}{m} \tag{3}$$

Here, $\dot{m}$ is the mass flow rate and is equal to $-\frac{dm}{dt}$ to define decreasing mass as a positive flow rate. After integrating both the right and left hand side and assuming constant thrust and mass flow rate the expression reduces to

$$\int_{v_i}^{v_f} dV = \int_{m_i}^{m_f} -\frac{T}{\dot{m}}\frac{dm}{m} = -\frac{T}{\dot{m}}\int_{m_i}^{m_f}\frac{dm}{m} \tag{4}$$

The left hand side of this equation is simply $v_f - v_i = \Delta V$. The right hand side is solved

$$-\frac{T}{\dot{m}}\int_{m_i}^{m_f}\frac{dm}{m} = -\frac{T}{\dot{m}}\ln(m)|_{m_i}^{m_f} = -\frac{T}{\dot{m}}\ln\left(\frac{m_f}{m_i}\right) = \frac{T}{\dot{m}}\ln\left(\frac{m_i}{m_f}\right) \tag{5}$$

At this point it is convenient to define the specific impulse of a rocket as $I_{sp} = \frac{T}{g_0\dot{m}}$, where g_0 is the gravitational acceleration at Earth's surface. The specific impulse is a measure of how efficiently, in terms of propellant consumption, a rocket is able to produce thrust. It quantifies how much thrust is produced for a given amount of mass flow. The higher the specific impulse, the more thrust is produced for a given mass flow rate, and therefore the more efficient the propulsion system. Using this in Equation 5 leads to

$$\Delta V = g_0\overline{I}_{sp}ln\left(\frac{m_i}{m_f}\right) \tag{6}$$

Here, $\overline{I}_{sp}$ is the average specific impulse. The term $\frac{m_i}{m_f}$ is sometimes referred to as the mass ratio, MR. Sometimes Equation 6 is written as

$$\Delta V = v_{eq}ln\left(\frac{m_i}{m_f}\right) \tag{7}$$

where v_{eq} is the engine's equivalent exhaust velocity. This can be seen by noting that a rocket's thrust is given by $T = \dot{m} v_{eq}$.

Equation 6 is known as the rocket equation. It was first derived by Tsiolkovsky in 1903 in the form shown in Equation 7 [167]. The rocket equation is a simple way to measure launch vehicle performance. Given an I_{sp} and the masses of the rocket, the total change in velocity that the rocket can impart can be calculated. This can useful in a wide variety of design studies. The change in velocity achievable by a rocket can be compared to the change in velocity required to reach orbital velocity, for example, or to perform a Hohmann transfer [50]. Alternatively, if the performance is fixed, the rocket equation can be used to size the rocket with a given payload mass. This type of performance analysis is very useful in early stages of launch vehicle design.

In terms of overall launch vehicle performance, ΔV is linearly proportional to specific impulse. It is intuitive that the more efficient a propulsion system is, the more velocity it will be able to impart for a given amount of propellant mass. The mass ratio is a measure of the structural system efficiency. The final mass term is essentially the initial mass minus the propellant-used mass. By minimizing structural mass, the final mass term is minimized, and thereby the natural-log term is maximized. The velocity change ΔV is exponentially related to the mass ratio. The total mass of a launch vehicle can be broken down into propellant mass, structural mass, and payload mass. Increases in structural mass directly affect the payload weight. In conceptual design a decrease in upper stage inert mass (or structural mass) has a 1 to 1 relationship to payload mass. Every pound saved is a pound gained for payload mass [177]. A more complex relationship exists for weight saved on other stages, but any weight saved will increase the payload. Once the vehicle is built, increases in required propellant weight will decrease the payload weight. In general launch vehicle design is driven largely by weight, because any increase in weight has an exponential effect on system performance.

Equation 6 above is sometimes referred to as the ideal rocket equation, or ideal ΔV equation, because it measures the maximum total velocity a vehicle can impart in an ideal environment. However, the ideal velocity change is never achieved. The assumptions made in Equation 2 are not applicable to a realistic launch environment. In reality, the derivation should start with

$$\sum F = ma = m\frac{dV}{dt} = Tcos(\phi) - D - G \tag{8}$$

Here, ϕ is the angle between the thrust vector and the velocity vector, D represents the aerodynamic drag force, and G represents gravity force. These terms include the losses experienced during the flight of a rocket that must be accounted for to determine the rocket's actual performance. Unfortunately there is no derived closed-form equation, like the rocket equation, that accounts for these losses in a launch trajectory for atmospheric launch. Instead they are included as lump terms, as seen in Equation 9 below.

$$\Delta V_{actual} = \Delta V_{ideal} - \Delta V_{thrust\ vector\ losses} - \Delta V_{drag\ losses} - \Delta V_{gravity\ losses} \tag{9}$$

This equation shows the actual performance as a function of the ideal performance of a rocket as well as some losses that result from flying the vehicle. These losses represent a degradation of the ideal change in velocity that the vehicle could achieve and are a function of the how the vehicle flies, or the vehicle trajectory. The three types of losses in Equation 9 are categorized as thrust vector losses, drag losses, and gravity losses [177]. Thrust vector losses are due to thrust vector and velocity vector misalignment. Basically, the thrust is being used for a purpose other than to linearly accelerate the vehicle. This can occur due to steering or imprecision in the thrust angle measurements of different engines. Drag losses are due to drag the vehicle experiences as it moves through the atmosphere. Drag is proportional to atmospheric density, which decreases exponentially with altitude. Drag losses can be minimized by gaining altitude as quickly as possible to get out of Earth's atmosphere. Gravity losses

are due to imparting a change in velocity against the acceleration of gravity. When thrust is perpendicular to gravity there are no gravity losses, because no thrust is being used to counteract gravity, and therefore all the thrust is imparting a change in velocity. Gravity losses are minimized by thrusting perpendicular to the gravitational force.

The path a launch vehicle flies to orbit is known as the trajectory. The three types of losses are all functions of the vehicle trajectory. Trade-offs exist between minimizing drag losses by gaining altitude and minimizing gravity losses by thrusting horizontally, all the while keeping steering angles small to minimize thrust vector losses. These trade-offs are exploited to find trajectories with few losses in a process called trajectory optimization. In real world applications, it is impossible to bring all these loss terms to zero, but the objective of trajectory optimization is to find the trajectory that minimizes the sum of all these losses while not violating any vehicle constraints, such as maximum acceleration or maximum dynamic pressure (used as an indicator of aerodynamic loads). Once this trajectory is found, the losses can be evaluated and the vehicle's actual performance ΔV_{actual} can be determined. Finding the best trajectory and evaluating losses is not a simple task. Equation 6 gives a closed-form solution of the ideal change in velocity a vehicle can impart. When designing a launch vehicle the actual performance in Equation 9 must be calculated, which requires computationally expensive numerical analysis [46].

1.1.2 The Cost of Space Launch

In addition to being challenging from a physics perspective, space launch is also a very expensive endeavor. In 2000, the average price for a pound of payload to geosynchronous orbit (GSO) was \$11,700 per *lb*, down from \$18,200 in 1990 [60]. For the Space Shuttle it was estimated that it cost around \$10,000 per *lb* to LEO [135]. The Atlas and Delta launch vehicles operate at a lower cost somewhere between \$3,000

and $5,000 per *lb* to LEO [135]. The least expensive estimate is for Russian and Ukrainian launch vehicles, at around $2,000 per *lb* to LEO [135]. From a simplistic perspective, every pound saved on propellant because of trajectory optimization can be translated to additional payload, worth between $2,000 and $10,000 per *lb* in orbit. Changes in the vehicle design or the trajectory design can result in a difference hundreds of pounds in the mass to orbit figure. This translates to hundreds of thousands of dollars.

An interesting side note is that as the launch vehicle market grows and demand increases, entrepreneurs are finding ways to cut costs and design, build, and launch spacecraft in more affordable ways. Elon Musk, for example, at Space Exploration Technologies believes that through manufacturing and process optimization the cost of launching to LEO can be decreased to less than $500 per *lb* [116]. Even at this price rate each pound in orbit is still significantly costly [164].

1.2 Launch Vehicle Design

Launch vehicle design is a complex process that requires detailed analyses in several disciplines including trajectory, weights/sizing, geometry, aerodynamics, heating, structure, and propulsion [159]. The analyses are highly coupled and can be characterized by thousands of design variables [29, 121]. When the disciplines are stitched together what results is a multidisciplinary environment representing the various aspects of a launch vehicle. Figure 1 shows an example of what this model environment might look like. Each box represents an individual discipline, and the lines represent information flow. This example is a re-creation of an environment described by Stevenson [163]. Many other versions and more detailed descriptions of multidisciplinary launch vehicle design environments can be found in the literature [22, 34, 79, 122, 132, 142, 179].

There are several ways to leverage the analysis tools for each discipline to complete

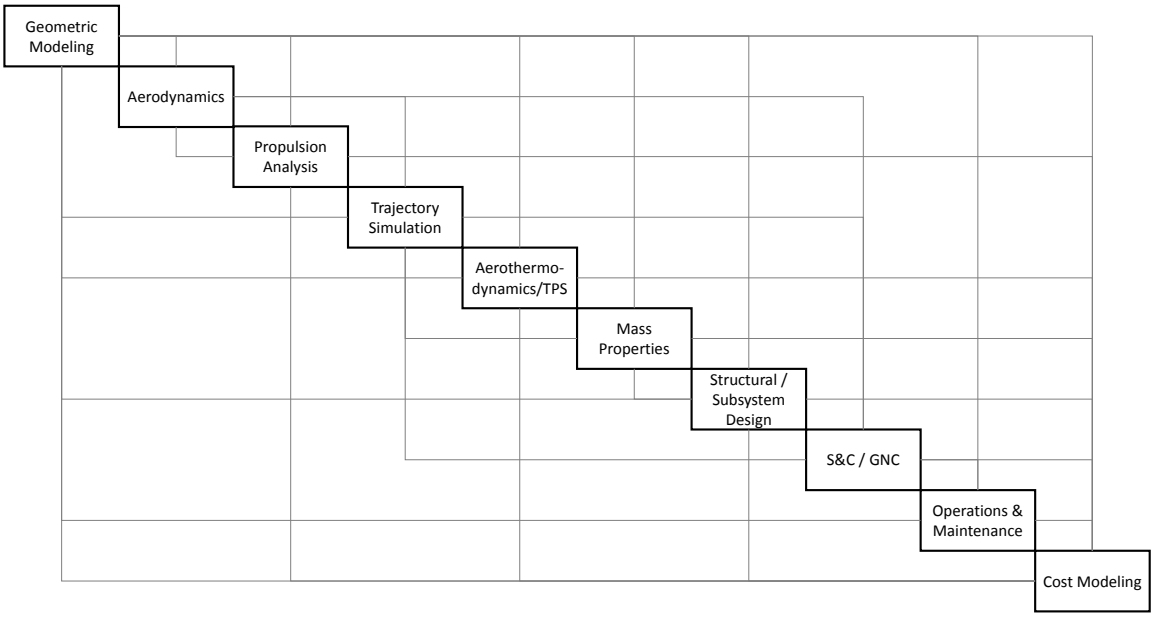


Figure 1: Sample Design Structure Matrix (DSM) for multidisciplinary launch vehicle design model [163]

a vehicle design. The traditional method involves breaking down the system design to a level where each part can be managed by a single engineer. Thus a group of engineers can individually work on a specific discipline and the contribution of all these disciplines will define the launch vehicle. This requires a system level oversight to make sure each of the parts are compatible and the overall requirements are met [22].

This process is used by the Earth to Orbit (ETO) team at the Advanced Concepts Office (ACO) at NASA Marshall Space Flight Center (MSFC). ACO provides preliminary design analysis for launch vehicles within NASA [176]. They break down the design into weights/sizing, trajectory, and structure/loads analysis. Engineers work in each of these areas until a solution is reached. One of the obvious downfalls of this method is that an engineer working on structural analysis must wait for the other analyses to be done before they can update the design. However, if multiple vehicles are analyzed at once, each engineer can perform his/her functions on one design as the other groups work on the other designs [176].

When the launch vehicle design problem is approached this way each discipline is individually optimized and compatible with the rest of the design. But the overall system is not necessarily the best solution [10]. Multidisciplinary design optimization (MDO) is a set of design methods that take into account the various interactions and couplings between a set of disciplines [156]. The goal of MDO is to find a design that, while not necessarily optimal from any single disciplinary perspective, is optimal at the vehicle level. Several different approaches to MDO exist, and many have been applied to aerospace systems [10, 29, 35, 36, 123, 141].

At the core of any launch vehicle multidisciplinary analysis model is trajectory analysis [34]. The reason trajectory analysis is so important is because it links the vehicle concept to vehicle performance. Vehicle performance is a major basis for comparison between vehicles. In conceptual design it is especially important to be able to down select between options and arrive at a few candidate vehicles. Unfortunately, trajectory analysis is a key problem in launch vehicle design, and is arguably the biggest challenge [9]. There have been multiple theses focusing on launch vehicle design that include efforts to deal with trajectory optimization [9, 34]. However, the problem of trajectory analysis in the context of vehicle design has not been solved. Consequently, this thesis will focus on the trajectory analysis aspect of launch vehicle design. With that in mind, the current approach for trajectory analysis is described in the following section.

1.2.1 Current Approach for Trajectory Analysis in Launch Vehicle Design

As with any design process, launch vehicle design can be framed using the generic top-down decision making process, shown in Figure 2 [111]. This process represents a generic approach to design. The first three steps are carried out to establish any requirements, identify stakeholders, consider any gaps and frame the problem.

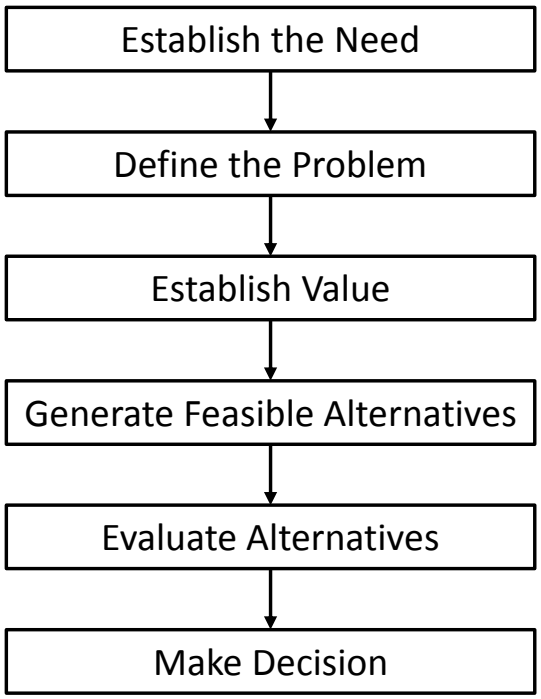


Figure 2: Generic top-down decision making process

The next two steps in the process are "Generate Feasible Alternatives" and "Evaluate Alternatives." In conceptual design it is desirable to consider as many options as possible to ensure the best designs are carried forward to the next phase. Figure 3 by Mavris illustrates this goal [112]. Design freedom represents the ability to consider many options through the analysis. The more options that can be considered, the more likely to find the best design concept. This correlates to higher design knowledge in the early phases of design. The motivation for this is that the cost that is committed by making design decisions is pushed later on in the design process when the best design has been selected. The obvious challenge of this goal is that it is not always simple to generate the design knowledge early on in the process. If design freedom is kept high, then it becomes even more of a challenge to generate the knowledge. In the generic decision making methodology, this problem arises when the alternatives that have been generated need to be evaluated. In conceptual design, the evaluation of alternatives in the context of trajectory analysis is carried out using a modeling and

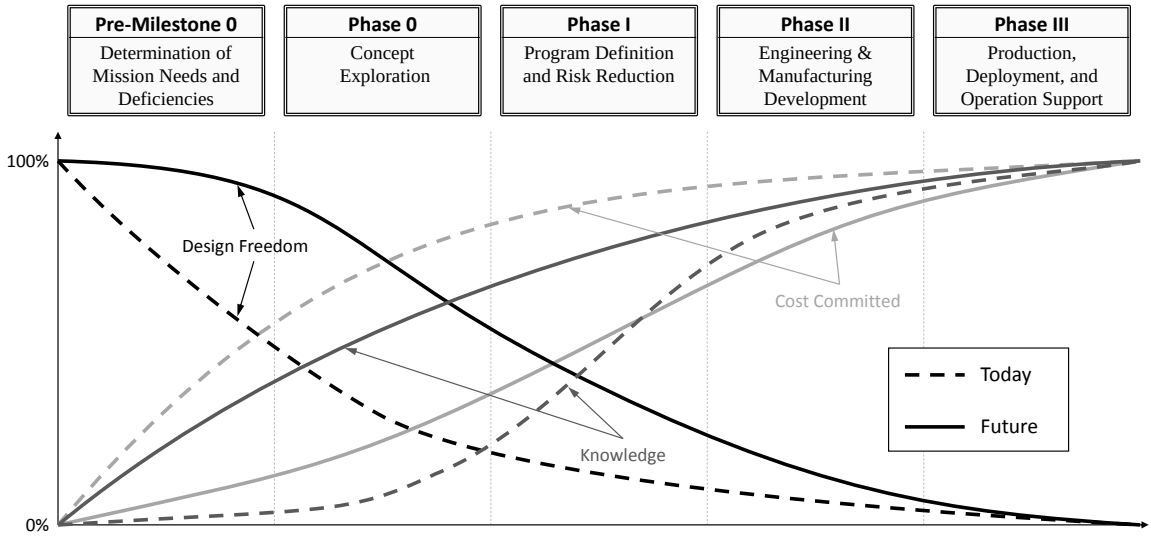


Figure 3: Design freedom, cost, and knowledge curves [112]

simulation (M&S) environment. This environment can come in many forms, from computer simulations to flight tests to expert opinion. In this case, flight testing is not a viable option and expert advice is not a feasible option. Computer simulation is used to evaluate the performance of launch vehicle alternatives in conceptual design.

Currently, the method used to perform the trajectory analysis is very time consuming and does not allow for the desired design knowledge. The method is heavily dependent on analysts and subject matter experts. Much of the work is done by hand in a serial manner. This severely limits the number of alternatives that can be evaluated. In a typical study, only a handful of vehicles may be evaluated.

The work in this thesis is aimed at developing an M&S environment that enables the evaluation of hundreds of thousands of alternatives, rather than merely a handful. Figure 4 shows where the contribution of this thesis would fit in the overall design process. Throughout this document, the current method will be described and the proposed method will be developed. The proposed method will rely heavily on current methods, but shift the focus from trajectory analysis to performance analysis. With this in mind, a discussion on the difference between performance and trajectory

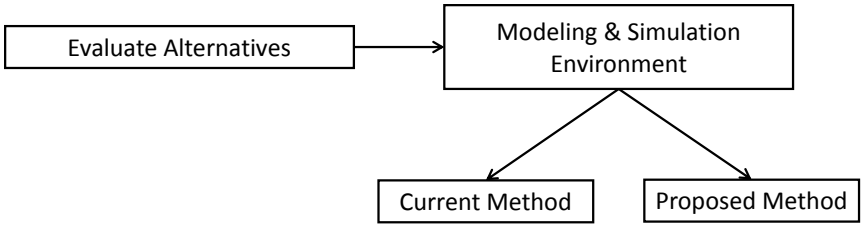


Figure 4: Proposed method in design process

analysis is presented next.

1.2.2 Performance Analysis vs Trajectory Analysis

The purpose of trajectory optimization in conceptual design is to provide the designer with the performance capability of the launch vehicle in question. Currently trajectory analysis is the only way of generating an accurate measure of the launch vehicle performance. Unfortunately, trajectory analysis is expensive. Consequently, in conceptual design, only a handful of vehicles are considered. Ideally thousands of vehicles would be considered, and the best chosen for further analysis.

Trajectory analysis, which is currently used in conceptual design, involves determining the set of controls that allows the vehicle to perform at the optimum, usually defined by payload delivered. The trajectory analysis determines the control variable values that result in the maximum amount of payload being delivered to the desired orbit for a given vehicle. This type of analysis maps a vehicle, defined by a set of design variables, to a trajectory and uses the trajectory to determine the performance. In other words, the performance of a vehicle is determined through trajectory analysis, even though the trajectory itself is not of particular interest in conceptual design. This is illustrated in Figure 5.

It is intuitive, however, that the vehicle performance is inherently tied to the design variables themselves. A vehicle with a better I_{sp} will perform better. The rocket equation is an example of this type of relationship. It relates launch vehicle performance directly to vehicle variables, namely the I_{SP} and MR, without considering

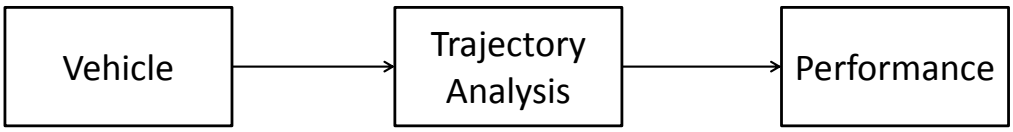


Figure 5: Launch vehicle trajectory analysis for performance evaluation

the trajectory. For the sake of argument, one can imagine the analysis possibilities if the rocket equation was used to evaluate vehicles in conceptual design. Thousands of different vehicles could be evaluated virtually instantaneously.

Figure 6 shows an example of the analysis capabilities. Ten thousand vehicles were analyzed in less than a second. The results were plotted with the ΔV performance metric on the x-axis. Probabilistic analysis is possible as well. As an example, assume a propellant mass is set to $50,000\ kg$ and the payload is $10,000\ kg$, but the I_{SP} and burnout mass are not fixed. Due to uncertainty in manufacturing or technology development, these values may not be determined yet. There is a distribution of values that they might be. A Monte Carlo simulation can be performed using the rocket equation. Distributions are assumed for the input variables, which are then sampled randomly. What results is a distribution on the performance. This can be shown directly, or as a cumulative distribution function (CDF), as seen in Figure 7. Essentially, the CDF shows the probability of meeting a certain performance given the input distributions. If a minimum ΔV of 5000 m/s is required, the CDF shows there is about a 5% chance of meeting this goal. Decision makers can then modify the requirements or introduce technologies to make the goal more feasible.

The examples given here are notional. They are meant to showcase the analyses possible in conceptual design with a performance evaluation method of the form of the rocket equation. This closed-form equation allows for incredible analysis power with the computational capabilities of an average computer. There is a significant problem with the rocket equation, however. It does not take into account any of the losses incurred during flight. If losses occur, Equation 6 is no longer valid. Indeed

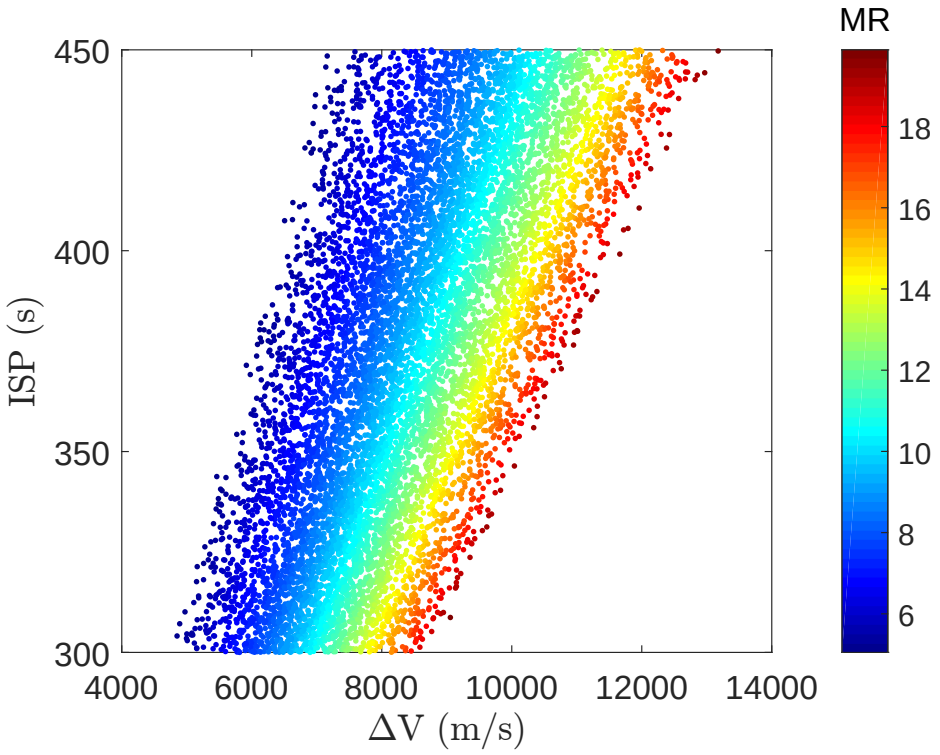


Figure 6: Launch vehicle concepts analyzed using ideal rocket equation

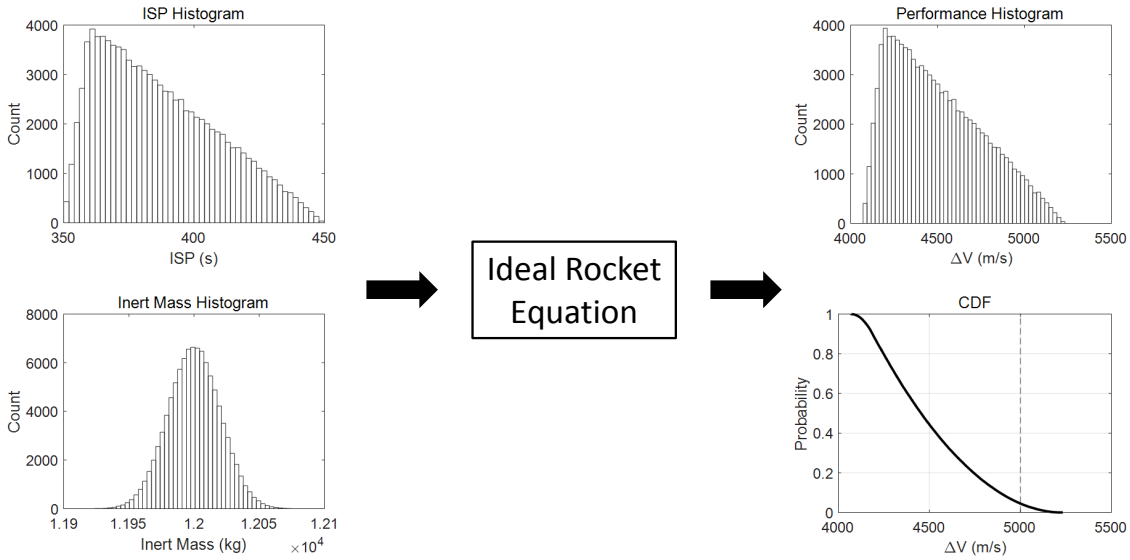


Figure 7: Notional probabilistic analysis using ideal rocket equation

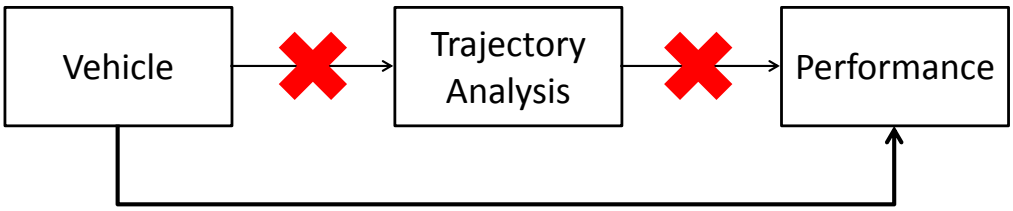


Figure 8: Launch vehicle performance analysis without trajectory analysis

the notional analyses shown in Figures 6 and 7 are not accurate. There does still exist a relationship between vehicle performance and the vehicle itself, but it is not as simple as Equation 6. It follows that for a given vehicle there is a given optimum performance, and if the vehicle changes, the optimum performance will change.

This represents a gap in the area of performance analysis. There is no accurate closed-form equation for launch vehicle performance analysis. On the one hand, the current method is time-consuming and labor intensive and limits the number of vehicles that can be considered conceptual design but provides accurate results. On the other hand the ideal rocket equation is fast but relies on simplifying assumptions and does not provide accurate solutions. A closed-form equation in the form of the rocket equation that takes into account losses incurred during flight would be a powerful tool for use in conceptual design.

The analysis that maps vehicle parameters directly to performance is referred to in this thesis as performance analysis, as opposed to trajectory analysis. It is illustrated in Figure 8. Using the rocket equation, as seen in the analysis shown in Figures 6 and 7 is an example of performance analysis. The beauty of performance analysis is that it does not involve the time consuming numerical integration and optimization process required in trajectory analysis. The analytical relationship can be evaluated for hundreds of thousands of vehicles in seconds using an average modern computer.

1.3 Research Objective

The comparison of the rocket equation to numerical trajectory analysis leads to the research objective. On one hand, the ideal rocket equation represents an analysis in the form of a close-form solution that allows for incredible analysis capability in conceptual design. This is of great importance as decision makers are analyzing alternatives and choosing the best options to carry forward. The availability of more information in this period of the design process means fewer changes later on, when they are significantly more costly. On the other hand, numerical trajectory analysis provides the accuracy needed to evaluate the actual performance of the launch vehicle. Without accuracy, any analysis is of little or no value.

This thesis will present a method that does not require trajectory analysis to evaluate each vehicle performance while still accurately accounting for relevant losses. Applications of this work will contribute to MDO, as well as general conceptual design of launch vehicles and even probabilistic analysis and technology evaluation for launch vehicles. With this in mind, the following research objective is given.

Research Objective

Enable rapid and accurate launch vehicle performance evaluation in the context of conceptual design.

The goal of this work is to provide a launch vehicle performance evaluation on-par with the current trajectory analysis in terms of accuracy but much faster. The current methods, discussed more in Chapter 2, are manual and time-consuming. The successful completion of this goal would allow designers to compare the performance of hundreds of thousands of vehicle concepts early in design and make informed decisions

as to what concepts to pursue. Essentially, the aim of this thesis is to reduce design cycle time with respect to launch vehicle performance analysis as a means to enable rapid and more vast design space exploration.

The development of this method will be the main contribution of this thesis. Other contributions will be identified throughout the thesis, but the overall goal is to develop a model that can be used in the future design of launch vehicles. This method provides great value to the launch vehicle conceptual designer.

1.4 Research Approach

Nothing has been said so far regarding whether the performance analysis is achievable or how to find the relationship between vehicle design variables and performance. This section will outline a strategy for developing a method to create this relationship between the launch vehicle and the performance. There may be a few concepts that are discussed here without full explanation or justification, but will be developed later on in the document. The purpose of this section is to introduce the proposed process so the reader will have the final goal in mind as each step is presented.

The key to the approach presented in this thesis for achieving the research goal is in finding the analytical relationship between launch vehicle design parameters and the launch vehicle performance. Ideally, this relationship could be found through analytical derivation starting from the fundamental laws of physics, as was shown with the rocket equation in Section 1.1.1. However, it is generally accepted in the literature that this is not a feasible approach for most problems [17, 20, 45, 173]. The only analytical derivations of performance analysis rely on significant assumptions that affect the accuracy of the results. Another approach is surrogate modeling, a statistical method that enables the generation of analytical relationships between inputs and outputs, in this case vehicle design variables and performance, in a data set based on the system of interest.

Surrogate models have been used in engineering analysis for over 60 years [26] and are especially useful for models that may be computationally expensive [11]. A surrogate model is created by fitting a statistical model to a set of data. The data is obtained by running the analysis environment of interest. The inputs are then mapped to the outputs via the surrogate model. What results is an analytical function that is used to approximate the analysis used to generate the data. It is important to understand that a surrogate model does not replace an analysis code. The model will only be as good as the data used to build it, so it is only valid within the ranges of data. But it does represent the data used to build it. A more detailed discussion of surrogate modeling is included in Section 3.7.

The method of surrogate modeling is a perfect fit for this problem. This is exactly what is needed to combine the benefits of a numerical trajectory analysis and a closed-form equation like the rocket equation. The accuracy comes from the data, which can be obtained using the accurate methods currently in use, while the form of the equation is provided by the surrogate model. Once the analytical relationship is created, hundreds of thousands of vehicles can be evaluated. The surrogate model can be used in a multidisciplinary optimization process, to perform a global optimization, or to assess probability of meeting certain requirements given future technology predictions.

1.5 Document Road-map

The proposed approach is to develop a surrogate model for use in conceptual design launch vehicle performance analysis. As seen in Figure 4, this method will represent the M&S environment used to evaluate the alternatives of interest. This surrogate model provides practically unlimited analysis capability. The overall process will be developed in depth and properly justified throughout the document. At this point a road map of the document is provided to help the reader understand each of the

steps as the method is developed.

A more detailed discussion on surrogate modeling is provided in Section 3.7. For now, it is sufficient to state that any surrogate model requires a set of data. For this application, the data consists of performance values for different launch vehicles. Chapter 2 provides a discussion on the current different methods available for generating this data and justifies the selection used for this thesis. The current methods alone, however, cannot be applied for the generation of data intended for surrogate models. There are various reasons for this that will be addressed. A sufficient optimization method is chosen for use in the rest of the process. Chapter 3 will introduce statistical methods that can be applied along with the chosen trajectory optimization method in order to successfully generate the data. Several steps in the proposed method are introduced and evaluated, ending with the creating of a surrogate model. Each step in the method will be experimentally tested and the results are presented throughout the document. In Chapter 4, the developed method will be applied to a sample problem and showcase using several types of conceptual design trade study activities. Finally, conclusions, contributions, and recommendations for future work are included in Chapter 5.

CHAPTER II

TRAJECTORY OPTIMIZATION

The earliest rockets did not require much in the way of trajectory optimization. As spaceflight developed, the ability to plan and carry out a specific mission became necessary, and launch vehicle trajectory optimization was born. There have been many techniques proposed and implemented over the years, and new techniques are still being developed today [19, 45, 65, 89, 158].

The goal of this chapter is to present the methods available for launch vehicle trajectory optimization that are relevant to the research objective of this thesis (in Section 1.3). Surrogate models were introduced in Chapter 1 as an enabler for meeting this goal. Creating a surrogate model requires data that maps the relevant inputs to the outputs. In this thesis, a trajectory optimization method will be required to generate this data. The different methods will be compared and contrasted in terms of how they approach the optimization problem as well as their applicability to this thesis. Advantages and challenges of each will be discussed with the end goal of selecting a trajectory optimization method to be used. Ultimately the data generated through trajectory analysis will be used to enable performance analysis, as discussed in Section 1.2.2. The chapter begins by presenting the general optimization problem and introducing some high level characteristics of the solution algorithms.

2.1 General Optimization Problem

The general optimization problem can be formulated mathematically as (as seen in *Multidiscipline Design Optimization*) [170]

$$
\begin{array}{lll}
\text{Minimize:} F(\mathbf{X}) & & \text{objective function} \\
\text{Subject to:} & & \\
\quad g_j(\mathbf{X}) \leq 0 & j = 1, m & \text{inequality constraints} \\
\quad h_k(\mathbf{X}) = 0 & k = 1, l & \text{equality constraints} \\
\quad X_i^l \leq X_i \leq X_i^u & i = 1, n & \text{side constraints}
\end{array}
$$

Where

$$
\mathbf{X} = \left\{ \begin{array}{c} X_1 \\ X_2 \\ X_3 \\ . \\ . \\ . \\ X_n \end{array} \right\} \tag{10}
$$

The design vector $\mathbf{X}$ represents variables in the user's control that determine the outcome of the objective function F. The objective function may be evaluated analytically, numerically, or even experimentally and yields some performance value. Inequality constraints are constraints where some function of the design variables must be less than zero. Equality constraints are similar, except that the function must equal zero. Side constraints are direct upper and lower bounds on the design variables themselves. In this formulation there are m inequality constraints, l equality constraints, and n side constraints.

2.1.1 Solution Techniques

A myriad of different solutions algorithms have been implemented for optimization [49, 54, 72, 90, 107, 155]. These solutions algorithms can be classified two different ways based on how they approach the problem. The choice of which kind of algorithm

to employ is very problem dependent, and the wrong choice can lead to long run-times and poor results [106].

2.1.1.1 Global vs Local Optimization

The first categorization is based on what kind of optimum the algorithm is designed to find, global or local. The global optimum is defined as the set of inputs which yields the best (either maximum or minimum) performance value over the entire design space [166]. In other words, of all the feasible combinations of design variables, the globally optimal set yields the best performance value. A local optimum is a set of inputs which yields the best performance value within a certain subsection of the design space [126, 178]. Figure 9 illustrates the difference between a global and local optimum. This one dimensional example function, given in Equation 11, is taken from Womersley [178].

$$F(x) = cos(14.5x - 0.3) + x^2 + 0.2x \tag{11}$$

An input value of $x \approx -1.1$, marked by the circle, is an example of a local optimum while $x \approx -0.2$, marked by the square, is the global optimum. An important observation is that a global optimum is a local optimum, but not all local optima are the global optimum [170].

An objective function to be optimized can be categorized as convex or non-convex [6]. Practically speaking, an optimization problem is convex if the local optimum is also the global optimum [101]. Figure 9, for example, is not a convex problem. If the input variable x were subject to the constrains $-0.3 \leq x \leq 0$, the problem would be convex. Determining if a function is convex or not, however, is not always straightforward, and for many real world problems, convexity cannot be assumed [166]. A non-convex problem implies there exists more than one local optimum in the region of interest; these problems are referred to as multimodal problems [166].

Local optimization, sometimes referred to as gradient-based optimization [134], is

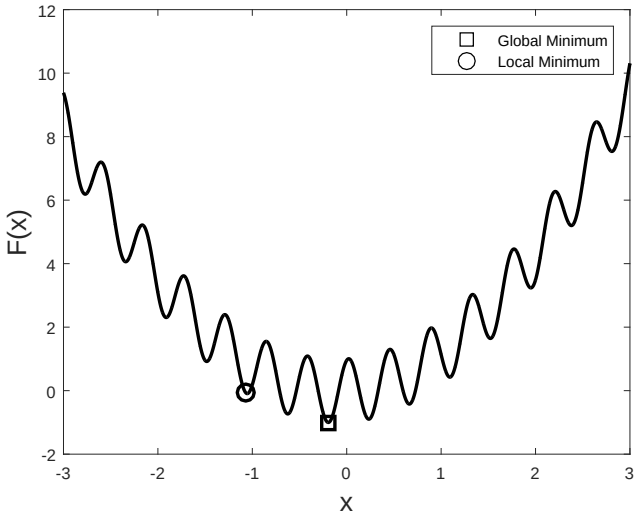


Figure 9: Example of global vs local optimum

a method that finds local optimum. As its name implies, it relies on information from the gradient to determine the optimum. The simplest example of a local optimization technique is the line search [170]. In one dimension, a starting point is selected, and steps are taken in a certain direction. At each step the objective function is evaluated. If the objective function improves, another step is taken; if not, a step is taken in the opposite direction. If steps in either direction worsen the objective function, the step size is reduced and the process is repeated. There are many ways to modify direction selection process and step size, and this leads to many different algorithms. Some of the more common ones included Steepest Descent Methods, Conjugate Direction Method, Simplex Method, and Sequential Quadratic Programming [170].

Local optimization techniques can be very powerful and are used in many optimization problems. By using gradient information, they are able to hone in on optima without wasting function calls exploring the entire area. For convex problems, local optimization methods are the best option [101]. For non-convex problems, however, local optimizers may get stuck in local minima [1]. For example, referring again to Figure 9, if the initial guess, or starting point, for the algorithm was chosen to be

$x = -1$, and the step size was small, a local optimizer would output $x \approx -1.1$, marked by the circle, as the optimum. An initial guess of $x = -0.3$, on the other hand, would return $x \approx -0.2$. The result of the optimization process is dependent on the initial guess. This is true for any local method in a multimodal problem. Different initial guesses can yield answers in different parts of the variable design space. This is not desirable and is the main disadvantage of local optimization methods.

Global optimization methods attempt to solve this problem by performing a search that is not entirely based on gradient information. They are designed to be able to find the global optimum even in highly multimodal problems [126]. A grid search, for example, is a global search that partitions the design space in some meaningful way and evaluate the objective function at each grid point. For discrete design variables, there exists the option to evaluate every feasible point. For continuous variables this is not possible (although a continuous space can be discretized). Global optimizers have the advantage that they explore the design space more exhaustively than a gradient based optimizer [101]. The opposite side of the coin, however, is that these methods generally require a much higher number of function calls. Recent advances in the area of computing, however, have helped overcome this hurdle to make global optimization methods more feasible and have led to a recent surge of research in this area [101]. Another disadvantage of global techniques is the inability to quickly find local optimum. Returning to the grid search example, an algorithm may pick the best performing points from an initial grid search and perform a smaller subsequent grid searches around these. The number of function calls using this method can quickly become prohibitively large.

A difference to note between local and global optimization techniques is that generally local optimization works with a single design vector $\mathbf{X}$ at a time. The optimizer modifies one candidate solution until it cannot improve further. Global methods usually work with populations of solutions. At each iteration in a global

method anywhere between 10 and 1000 candidate solutions may be evaluated, and information from all of those may be used to determine the candidates for the next population. The output, then, of a local optimization method is a single solution, while the output of a global method can be either a single solution or a family of solutions.

2.1.1.2 Deterministic vs Stochastic Optimization

The second categorization of optimization methods is based on the use of random numbers. An algorithm is categorized as deterministic if for a given set of inputs, the output is always the same. The term deterministic is defined more formally in Liberti [101], but the above definition will suffice for the purposes of this discussion. Stochastic algorithms use random numbers to generate the output. This means for a given input, the output will vary. The variation in the output is due to the random nature of the algorithm, not necessarily to a multimodal problem as is seen with local methods in Section 2.1.1.1. At first glance this may seem of little value, but stochastic methods have been used extensively in optimization [46, 127]. A more in depth discussion of how stochastic methods are used in trajectory optimization is included in Section 2.3.3. An interesting note is that computer algorithms are inherently deterministic. Therefore, to generate random numbers, pseudo-random number generators are used to simulate randomness [101].

One matter of importance regarding stochastic algorithms is the issue of convergence. Convergence in this context refers to how well the stochastic method finds global optimum [166]. Stochastic algorithms are based on probability, and therefore convergence is not guaranteed for anything less than an infinite number of cases if any of the control variables are continuous. This is not feasible, so these algorithms must be generated in such a way as to maximize the probability of convergence. One way to do this is simply to run a specific case multiple times [124].

The two categorizations, global vs local and deterministic vs stochastic, are independent and can be combined in any way. A global method, for example, can be either deterministic or stochastic, and there exist both local and global stochastic methods. However, most local methods are deterministic, and global methods are generally stochastic [101]. This is because local methods inherently have a path to follow based on gradient information, which is defined by the objective function and constraints. Global methods, while trying to explore the whole space, attempt to spread out and look at areas that may not seem promising, and random numbers can be used to achieve this.

2.2 Optimal Control Problems

An optimal control problem (OCP) is a type of optimization problem where the goal is to determine "the inputs to a dynamical system that optimize (i.e., minimize or maximize) a specified performance index while satisfying any constraints on the motion of the system" [134]. The dynamical system is defined by a set of ordinary differential equations (ODE's). The state of the system at any time is found by solving the set of differential equations given initial conditions, optimization parameters, and control functions. The state may be subject to path constraints, which limit how the system can behave, and boundary conditions, which determine the state of the system at the initial and final times. Stated mathematically (as formulated in Betts [19])

$$
\begin{aligned}
&\dot{\mathbf{y}} = \mathbf{f}\left[\mathbf{y}(t), \mathbf{u}(t), \mathbf{p}, t\right] && \text{system equations} \\
&\boldsymbol{\psi}_{0l} \leq \boldsymbol{\psi}\left[\mathbf{y}(t_0), \mathbf{u}(t_0), \mathbf{p}, t_0\right] \leq \boldsymbol{\psi}_{0u} && \text{initial boundary conditions} \\
&\boldsymbol{\psi}_{fl} \leq \boldsymbol{\psi}\left[\mathbf{y}(t_f), \mathbf{u}(t_f), \mathbf{p}, t_f\right] \leq \boldsymbol{\psi}_{fu} && \text{final boundary conditions} \\
&\mathbf{g}_l \leq \boldsymbol{g}\left[\mathbf{y}(t), \mathbf{u}(t), \mathbf{p}, t\right] \leq \mathbf{g}_u && \text{path constraints} \\
&\mathbf{y}_l \leq \mathbf{y}(t) \leq \mathbf{y}_u \quad \mathbf{u}_l \leq \mathbf{u}(t) \leq \mathbf{u}_u && \text{bounds on state and control variables}
\end{aligned}
\tag{12}
$$

where $\mathbf{y}(t)$ is the state vector, $\mathbf{u}(t)$ is the control vector, $\mathbf{p}$ is the static parameter vector, and t_0 and t_f are the initial and final times respectively.

OCP's pose a challenge in that the elements of the control vector $\mathbf{u}(t)$ are functions instead of parameters. The general optimization problem presented in Section 2.1 is often referred to as a nonlinear programming (NLP) problem [170]. The goal of an NLP problem solver is to find values for the finite number of parameters so as to minimize the objective function. The goal of an OCP solver is to find the control *functions* to minimize some cost function. NLP solvers work with parameters, where OCP solvers work with functions. One way to understand the difference is to view an OCP as an NLP problem with infinite number of variables [20], where these variables define a function.

In practice, it is much more difficult to solve OCP's. Because optimal control problems are infinite-dimensional, the methods that apply to NLP problems cannot be directly applied to OCP's. The following sections will describe the methods that have been developed to solve OCP's.

2.2.1 Optimization Approach for Optimal Control Problems

A NLP problem can be solved using methods from calculus. The first and second derivatives of a function can be used to find the necessary and sufficient conditions for optimality. When the function under consideration is convex, this leads to the global optimum. For example, minimizing the function x^2 leads can be differentiated twice to yield $x' = 2x$ and $x'' = 2$. Setting the first derivative equal to 0 and solving leads to the solution $x = 0$, and the second derivative being positive shows that this point is a minimum. If the derivatives cannot be found globally for a function, the local derivative is used to inform which direction to search in. Methods for the solution of NLP problems are generally based on this approach of using derivative information [18]. The calculus used here applies to variables. When solving OCP's, however, the

solver works with functions instead of variables. This requires functional calculus, or calculus of variations [41]. This leads to a different approach for solution algorithms.

The ideal solution approach to an OCP would be analytic. In other words, the solution could be mathematically derived from the problem statement without the need for a numerical algorithm. The example in the previous paragraph is an analytic solution using classical calculus. An example of an analytic solution to an OCP is shown later in the document in Section 2.6.1. Unfortunately, this is rarely possible for real world problems [45]. Instead, numerical methods are required for most OCPs [140, 164].

There are two essential parts to the solution technique for an OCPs [20, 134]. The first is a method to solve the set of differential equations that represent the behavior of the system. The discussion on these methods is deferred until Section 2.2.2. The second and more obvious part is the optimization philosophy, which defines how the problem is formulated. In general there are two categories: direct and indirect [17, 19, 20].

2.2.1.1 Direct Methods for Optimal Control Problems

Direct optimization methods solve the infinite dimensional OCP by converting it into a finite dimensional NLP problem in a process called transcription [20]. The transcription process involves a discretization of the control parameters and/or the state parameters [78]. Equation 13 shows how a control function $u(t)$ might be discretized [164]. The function is approximated by a set of parameters. Each parameter, u_1 through u_n, applies during a certain phase. In this example the phases are defined by time.

$$u(t) = f(t) \rightarrow u(t) = \begin{cases} u_1, & t_1^i \leq t < t_1^f \\ u_2, & t_2^i \leq t < t_2^f \\ & . \\ & . \\ & . \\ u_n, & t_n^i \leq t < t_n^f \end{cases} \tag{13}$$

The parameters $u_1, ..., u_n$ are applied as the control during the time periods defined by $t_1^i, ..., t_n^i$ and $t_1^f, ..., t_n^f$. The parameters $u_1, ..., u_n$ themselves may be constants or they may be functions defined a finite set of parameters. For example, $u_2 = u_{2,0} + t \times u_{2,1}$. In this case the control u_2 is described by a constant term $u_{2,0}$ and a rate $u_{2,1}$. In general, any function can be employed. The parameters required to define the function become the design variables in the NLP problem. Figure 10 is a pictorial depiction of trajectory and control discretization. In both Equation 13 and Figure 10 the trajectory was discretized using time. In general, phases can be defined by any variable, but care must be taken to ensure the approximation of the control function is well-defined.

When a problem is transcribed, the resulting NLP problem inherently has fewer degrees of freedom than the OCP. In fact, this is the very reason problems are transcribed. However, by doing this, solutions become sub-optimal [88]. The control function is being approximated by some function, and hence this approximation leads to sub-optimal solutions to the OCP, even if the NLP problem solution is itself optimal. Another consequence is that problems solved using the direct method can be multimodal, where local optima will exist in different parts of the design space. This results in a situation like the one illustrated in Section 2.1.1.1, where the optimization result is dependent on the initial guess for the design variables.

Direct methods have the advantage of being relatively robust (compared to other

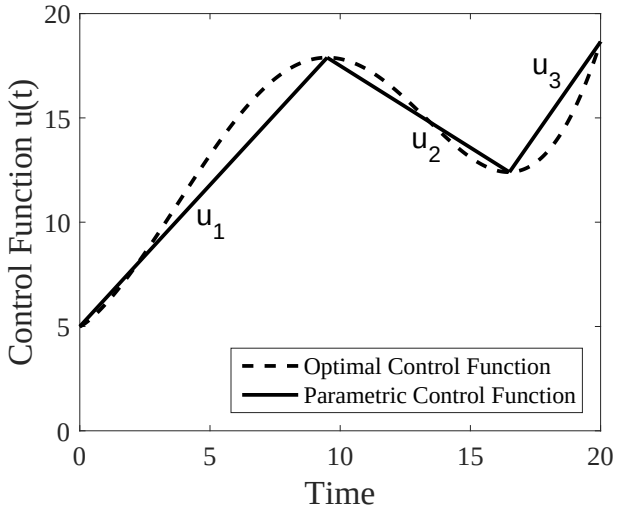


Figure 10: Example of control discretization

local methods) and relatively simple to implement [45]. However, they are less accurate [148] than the second branch of local optimization methods: indirect methods.

2.2.1.2 Indirect Methods for Optimal Control Problems

Indirect methods solve the optimal control problem using calculus of variations. A solution is obtained by deriving necessary conditions. An augmented performance index is created using Lagrange multipliers or costates to include the constraints [88]. This results in a boundary value problem [164]. An optimal control function is derived based on the dynamic system and the constraints. Indirect methods solve for a control at each point in time by defining a function for the control that can be solved at any point. The functions that result from solving and OCP indirectly is known as a guidance laws [61]. A famous example of a launch vehicle guidance law is the linear-tangent steering law [125, 177]. This guidance law is implemented in an example later on in the document in Section 2.6.1.2.

Indirect methods are very powerful, and yield very accurate results. However, deriving the necessary conditions for complex systems can be very difficult. In addition,

these necessary conditions are unique to each problem, so developing a tool for general trajectory optimization is nearly impossible [173]. Finally, because the costates are included as part of the optimization process, the number of optimization variables increases. Initial guesses for costate values are difficult to obtain because they have no physical meaning. One solution is to randomly guess different initial values for the costates until a solution is found [61]. However, this can lead to long run-times. Indirect methods have been studied significantly, and the general consensus on these methods is summed up in the following quote from Bryson [32]:

> "The main difficulty with these methods is *getting started*; i.e., finding a first estimate of the unspecified conditions at one end that produces a solution reasonably close to the specified conditions at the other end. The reason for this peculiar difficulty is the extremal solutions are often *very sensitive* to small changes in the unspecified boundary conditions... Since the system equations and the Euler-Lagrange equations are coupled together, it is not unusual for the numerical integration, with poorly guessed initial conditions, to produce "wild" trajectories in the state space. These trajectories may be so wild that values of $x(t)$ and/or $\lambda(t)$ exceed the numerical range of the computer!" [31]

Because of this, indirect methods have not been widely implemented in many general trajectory optimization problems [19, 45].

2.2.2 Numerical Integration Methods

Regardless of how the trajectory optimization problem is solved, whether via direct or indirect methods, the differential equations must be solved [74]. As stated earlier, analytic solutions for the differential equations of problems of this complexity do not exist, so numerical methods must be employed [164]. In trajectory optimization, the first of two main methods is called shooting, or time-marching.

2.2.2.1 Shooting Method

Shooting calculates the current state based on information from either current or previous state information. Essentially, at each time step the system equations are calculated, and the resulting derivatives are used to update the state to the next time step. There are several techniques on how the derivatives are used to update the state. Euler methods are shown in Equation 14

$$\mathbf{x}_{k+1} = \mathbf{x}_k + h_k \left[\theta \mathbf{f}_k + (1-\theta)\mathbf{f}_{k+1}\right] \tag{14}$$

Where $\mathbf{f}_k = \dot{\mathbf{x}}_k$ and h_k is the time step. When θ is 1, the method is called Euler forward, because the next state is dependent entirely on the information from the previous state. This type of numerical integration is called explicit integration [8]. When θ is 0, the Euler backward method is used [7]. In this case the next step is dependent on the previous state values but derivatives from the next state. These methods are called implicit integration methods because the state x_{k+1} is on both sides of the equation [8]. Because of this, the equations must be solved iteratively. In general, explicit methods are easier to implement and more computationally efficient, but not as accurate as implicit methods [70].

Euler methods are the simplest form of shooting methods [165]. Probably the most common numerical integration method, however, is an explicit fourth order Runge-Kutta method [134]. This method is shown in Equation 15 below.

$$\begin{aligned}
\mathbf{k}_1 &= h_i \mathbf{f}(\mathbf{x}_i, t_i) \\
\mathbf{k}_2 &= h_i \mathbf{f}(\mathbf{x}_i + \frac{h_i}{2}\mathbf{k}_1, t_i + \frac{h_i}{2}) \\
\mathbf{k}_3 &= h_i \mathbf{f}(\mathbf{x}_i + \frac{h_i}{2}\mathbf{k}_2, t_i + \frac{h_i}{2}) \\
\mathbf{k}_4 &= h_i \mathbf{f}(\mathbf{x}_i + h_i \mathbf{k}_3, t_i + h_i) \\
\mathbf{x}_{i+1} &= \mathbf{x}_i + \frac{1}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4)
\end{aligned} \tag{15}$$

The two methods discussed thus far, Euler and Runge-Kutta, are single-step methods, because only one point is used to compute the second point (even though implicit

methods require the current and previous point). There are several other types of single-step methods, such as Heun and Taylor methods [108]. Another class of numerical methods use several of the previous steps (once the algorithm has been started) to compute the next step. These are known as Predictor-Corrector methods. Predictor-Corrector methods are generally more complex, but can be more accurate. One example used in spaceflight trajectory optimization is the Adams-Bashforth-Moulton method [14]. For this study, single-step methods are used because those are the methods implemented in current trajectory optimization tools, as will be discussed in Section 2.3. The reader is referred to Mathews [8] and Atkinson [108] for a more in depth discussion on these and other numerical methods for the solution of ODEs.

2.2.2.2 Collocation Method

In literature, collocation is sometimes referred to as transcription [19, 78]. Note, the word transcription is used differently when speaking in the context of direct optimization methods, discussed in Section 2.2.1.1. For the purposes of this discussion, transcription will be used only in the context of direct methods, and collocation will be used to refer to the method of numerically solving differential equations.

Collocation employs an interpolating function to approximate the state of the system. Usually the interpolating function is a polynomial. At collocation nodes, constraints are used to compare the derivative of the approximating function to the solution of the system of equations at that point. These constraints are called defect constraints and are shown in Equation 16.

$$\xi = \mathbf{X}(t_j) - \mathbf{f}(\mathbf{x}(t_j), t_j) \tag{16}$$

Figure 11 below illustrates the collocation method. Frank [59] summarizes the collocation method as construction of "a polynomial that passes through y_0 and agrees with ODE at s nodes on $[t_0, t_1]$. Then [...] let the numerical solution be the value of this polynomial t_1." In this context the time step between t_0 and t_1 is broken up

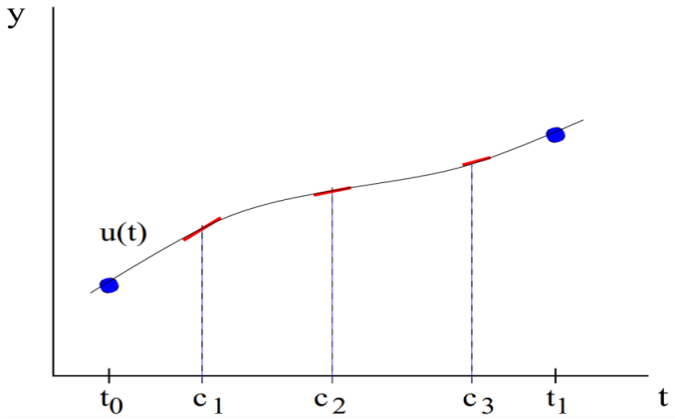


Figure 11: Collocation method [59]

into s nodes referred to as $c_1, c_2, ..., c_n$. The red lines in Figure 11 represent the slope of the polynomial, which is compared to the numerical solution of the system equations. Because of the way the collocation method solves differential equations, namely solving for all the variables at once, it is considered an implicit method.

2.3 Launch Vehicle Trajectory Optimization

The problem of finding the best trajectory to orbit for a launch vehicle is an example of an OCP. The equations of motion for a vehicle moving over the earth are the set of ordinary differential equations governing the system. There are control variables the affect the behavior of the system, such as launch azimuth. The variable used to control how the vehicle flies throughout the trajectory, the pitch rate for example, is a function $u(t)$. Up to this point the discussion has focused on general solution methods to OCP's. This section will present some of the most relevant methods used specifically for launch vehicle trajectory optimization. The local optimization methods will be discussed first. Recall that the goal is to find a trajectory optimization method that can be applied to generate data for the creation of a surrogate model.

2.3.1 Local Methods

In local trajectory optimization the combination of the optimization approaches from Section 2.2.1 and the numerical integration methods in Section 2.2.2 leads to four main types of methods for trajectory optimization. A fifth method is also considered, even though it is only an extension of two of the four main methods. A brief discussion of each method is included to investigate benefits and challenges as well as discuss any general purpose tools that have implemented these methods.

2.3.1.1 Direct Shooting

Direct shooting is the easiest of these methods to understand and visualize. As a direct method, the control function is represented by a finite set of parameters [45]. A cost function, which can include terms based on path and final constraints, is evaluated by numerically integrating the equations of motion, explicitly or implicitly, given initial and ending conditions. The control parameters are then modified based on gradient information to improve the cost function [134]. Optimization algorithms are techniques on how to modify the control parameters, and there are many different algorithms that exist [170].

Direct shooting methods work well when the control function can be approximated by a small number of parameters. The ascent of a launch vehicle to orbit is a good example of this [19, 173]. Because gradients are calculated for each parameter, as the number of parameters increases the process becomes computationally expensive. Solving problems that required a large number of parameters may become intractable. Gradient calculation can be another problem for the direct shooting method. Numerical issues can lead to inaccurate gradient calculations. In addition, small changes in control parameters can lead to extremely non-linear behavior in system constraints. This makes it very difficult for an optimizer to solve the problem [19].

Despite these difficulties, direct shooting is the most common method in trajectory optimization. Program to Optimize Simulated Trajectories (POST) is a tool developed by NASA Langley Research Center and Lockheed Martin Astronautics in the 1970's. It is used to calculate trajectories for air-breathing or rocket ascent and reentry in arbitrary environments [28]. POST has three optimization algorithms built in. The first two are the un-accelerated and accelerated projected gradient method (PGA) [128]. These only use first order gradient information. The third algorithm is an optimization package NPSOL developed by the Systems Optimization Lab at Stanford University [128]. NPSOL employs second order gradient information, which can be more accurate, but generally takes longer. For more information about the formulation of POST the reader is referred to the POST Formulation Manual [27] or to Brauer [28].

Since the original version, there have been several upgrades to POST. Some of the notable ones include adding 6 degrees of freedom as well as the capability to simulate multiple vehicles at once.

2.3.1.2 Indirect Shooting

Like direct shooting, indirect shooting uses numerical integration, either explicit or implicit, to solve the set of ODE's. However, instead of discretizing the control function, the necessary optimality conditions are derived based on a cost function augmented with Lagrange multipliers and constraints [61]. This leads to a set of optimization variables including simulation time and Lagrange multipliers, or costates. If path constraints are included, they are dealt with by discretizing the trajectory into phases based on whether they are constrained or unconstrained and solving the individual phases [19]. This will increase the number of design variables.

There are several complications that arise when using indirect shooting. As with all indirect methods, the necessary conditions must be derived. This can become very

complicated. A program like POST allows the user to select from multiple reference frames and different environment models. Deriving the necessary conditions while allowing different reference frames and environment models would be an arduous task. Because of that most indirect shooting tools are considered somewhat inflexible [74], and only useful for a small set of specific problems. There is work being done to overcome these challenges [19], but direct methods have been developed to a point where indirect methods do not seem as necessary [20, 46].

2.3.1.3 Direct Collocation

Collocation methods were first implemented in indirect optimization, but then applied to direct methods to remove the requirement of deriving the necessary conditions. In direct collocation the set of ODE's is replaced by a set of defect constraints (see Equation 16) at grid points. When formulating a problem in this way the number of variables increases dramatically. The set of optimization variables includes state and control variables at each grid point as well as initial and final conditions. A problem may have on the order of thousands of optimization variables. Calculating the required matrices in the NLP problem proves costly. However, for direct collocation problems, about 99% of the entries of these required matrices are 0, and therefore algorithms are used to take advantage of the matrix sparsity to reduce computational costs [19].

When set up correctly, direct collocation can be a very powerful method for optimizing trajectories. For this approach to be efficient, however, matrix sparsity must be taken advantage of, and this can be difficult to implement. The method of direct collocation has been implemented in a tool called Optimal Trajectories by Implicit Simulation (OTIS). OTIS was developed in the mid 1980's by NASA Glenn Research Center and The Boeing Corporation. OTIS was designed to optimize trajectories for many types of vehicles, including launch vehicles, satellites, and aircraft. While

OTIS has an option to use the shooting method to solve the ODE's its power lies in the collocation method. The goal of OTIS was to produce a general purpose trajectory simulation and optimization analysis tool. Like POST, OTIS is integrated with optimizer packages. One of the original ones was Boeing's Chebyshev Trajectory Optimization Program (CTOP) [74]. In the latest version a more powerful optimizer is used: SNOPT 7. SNOPT 7 is a sparse non-linear programming optimizer developed at the Systems Optimization Lab at Stanford University. It is designed to take advantage of the matrix sparsity of these type of problems [115]. A more complete description of OTIS is given by Hargraves [74].

Currently OTIS and POST, discussed previously, are the two main trajectory optimization tools used in industry. Opinions regarding which tool provides better results are generally based on which tool the user is more familiar with. The underlying physics in both tools, however, is the same. It has been shown, that there is little numerical difference in the results calculated by POST and the results calculated by OTIS for a given problem [119].

It is interesting to observe that both major tools used in industry apply direct optimization techniques. This is no coincidence. The inherent issues with indirect optimization make it difficult to easily apply to general trajectory problems. Therefore, the tools that have been adopted by industry as standard all employ direct techniques.

2.3.1.4 Indirect Collocation

Indirect collocation was developed to solve necessary conditions for boundary value problems using a different numerical solution technique. Indirect collocation methods have been used in many different applications. However, these methods suffer from many of the same problems as indirect shooting methods, and therefore have not been implemented for general purpose trajectory tools.

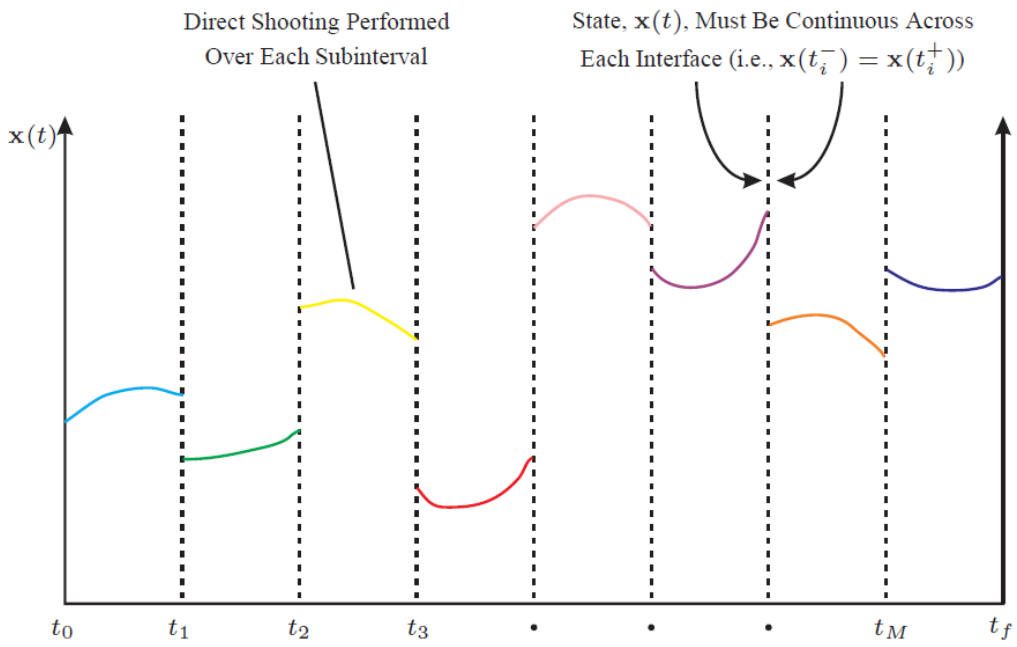


Figure 12: Schematic of direct multiple-shooting method [134]

2.3.1.5 Multiple Shooting

When shooting methods are employed, small changes in control variables early in the trajectory have a large effect on the trajectory's ending state. This is due to long simulation times. A solution to this is to employ a multiple shooting method. Essentially, the idea is to perform a shooting based optimization, either direct or indirect, for segments of the total trajectory, and enforce continuity constraints at the phase boundaries. It is similar to the collocation method, in that the equivalent to a defect constraint is introduced to ensure the simulation is physically feasible. Figure 12 shows the direct multiple shooting method as an example.

Multiple shooting does increase the number of variables needed to solve the system. However, even with this increase in number of variables, multiple shooting methods can be an improvement to single shooting methods. In general, each phase of a multiple shooting method will have the disadvantages associated with the single shooting method employed, either direct or indirect [134].

Table 1: Summary table of local optimization methods

		Optimization Method	
		Direct	**Indirect**
Differential Eqn. Method	**Shooting**	Direct Shooting: • approximates optimal control problem as NLP • employs explicit or implicit integration	Indirect Shooting: • solves necessary conditions to obtain control values • employs explicit or implicit integration
	Collocation	Direct Collocation: • approximates optimal control problem as NLP • approximates system states with polynomials	Indirect Collocation: • solves necessary conditions to obtain control values • approximates system states with polynomials

2.3.2 Local Methods Summary

This section described four main types of local methods used in launch vehicle trajectory optimization, which are summarized in Table 1. General purpose trajectory optimization tools have been implemented using two of these methods: direct shooting and direct collocation.

From literature the general consensus is that direct methods are much easier to work with. Conway states that indirect methods display a "lack of robustness" [46]. Ross et al. state that "direct methods are preferred over indirect methods" [140]. Seywald states that "rapid trajectory generation is usually attempted by applying direct optimization techniques" [145]. There are several other examples from literature that point to direct methods as easier to implement and work with [133, 173]. As far as the author can tell, no one supports indirect methods as the solution of choice for a general trajectory optimization process, especially for use in conceptual design.

There are several reasons indirect methods are generally hard to work with. Indirect methods require an initial guess of costate variables, which have no physical meaning and are therefore hard to guess [145]. It can be hard to include tabulated data for aerodynamics, atmosphere, or propulsion parameters [46]. Finally, and likely the biggest difficulty, any change in the problem requires a re-derivation of the necessary equations. This is a huge obstacle if different launch vehicle concepts are being considered, especially if the process is to be automated.

Direct methods are generally preferred, but they are not without drawbacks. These methods are generally less accurate, and can have errors of 1% [164]. Direct methods also require initial guesses. It is easier to find a feasible initial guess for a direct method than an indirect. Recall from Section 2.2.1.1, a problem that arises with the initial guess for direct methods is that the optimization algorithm may find a local minimum that is near the initial guess instead of the global minimum [1, 46, 164]. For launch vehicle trajectories optimization using the direct method there are many local minima, and a direct method optimizer finds solutions that while locally optimal, are not always the best global solutions [162].

2.3.3 Global Methods

Global methods have not always been popular in trajectory optimization. Bett's survey paper in 1998 gave a detailed review of the then current trajectory optimization techniques and referred to global methods as simply not worth it. In a discussion about genetic algorithms specifically he states

> "Unfortunately, because they do not exploit gradient information, they are not computationally competitive with the methods in Sec. IV. [local methods]." [19]

In the late 1990's, however, there was a significant increase in state-of-the-art computational performance. This led to a surge of research in the area of global optimization methods [101]. In a more recent survey (2012) on trajectory optimization, Conway describes global methods as having advantages over the direct and indirect methods described in Section 2.3.1 [46].

When global methods are used, the trajectory problem must still be represented using a finite number of parameters for global optimization to be useful [46]. For indirect methods this means using variables with no physical meaning, making them difficult to estimate and set ranges on (see Section 2.3.2). Consequently, the trajectory problem is usually transcribed, and the global optimizer can operate on a finite number of control parameters.

When global techniques are employed, they are usually coupled with a local optimizer to fine tune candidate solutions. The global technique is used to exhaustively explore the design space and find a family of candidate solutions. Once solutions of interest are found, the local technique is used to refine the solutions [35, 46, 101]. The integration of the local and global methods can take many forms [1].

Grid searches and random searches are two simple global methods. Whether they are optimization methods or search methods is a matter of debate, and will not be explored here. For the purposes of this discussion they will be referred to as optimization methods. Grid searches allocate a certain number of points to each variable in the design space and evaluate the objective function at each combination of points. Random searches randomly select points inside the design space to evaluate the objective function at. Figure 13 illustrates grid vs random search.

As far as the author can tell, only one tool, QuickShot, has been developed specifically for launch vehicle trajectory optimization using global methods. Global methods have been employed for trajectory optimization problems, but no tool developed. QuickShot is a tool that has been developed by SpaceWorks Enterprises, Inc. with

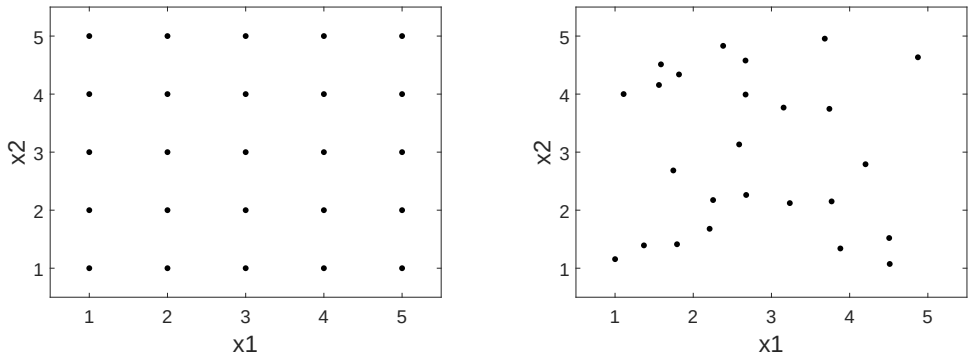


Figure 13: Grid vs random search

the purpose of decreasing the number of trajectory assumptions, decreasing the need for a good initial guess, and avoiding the need for an expert user in the loop [64]. Global methods provide all these advantages at the cost of increased computational requirements. QuickShot was validated against POST to show similarity of optimized trajectories. The global search methods employed in QuickShot are the grid and random searches. The best results from the global search are then input into a local search to find the local optimum. A schematic of QuickShot is shown in Figure 14.

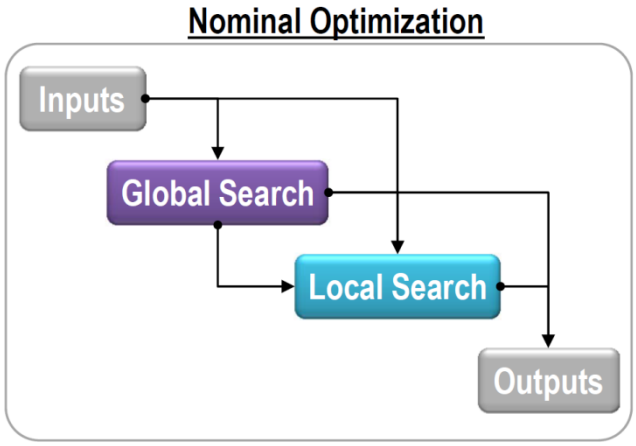


Figure 14: QuickShot optimization method [64]

In the following sections an overview of some of the most common global methods is given. This is a relatively new field in trajectory optimization, and many methods

are being employed and updated. This discussion by no means includes all the options, but briefly describes some of the more well-known methods. One note before continuing is that global methods can use either collocation or shooting to numerically integrate the trajectories.

2.3.3.1 Genetic Algorithm

Genetic Algorithms (GAs) were developed to model the theory of evolution. They are inherently designed to work with discrete design variables, but can be easily modified to accommodate continuous variables [170]. They way GAs work is by representing a point in the design space by a binary string. Operations, such as reproduction, crossover, and mutation, are performed to modify the binary string, which leads to a new point in the design space. The operations performed are designed to only let the best solutions survive (survival of the fittest).

A basic GA search is performed in the following way. A number of random points are chosen to initialize the algorithm. The set of current solutions is known as the population, and a single solution is called a member. A new population is generated every iteration. The series of populations are termed generations. The performance index is evaluated for all members. Members of the population are randomly chosen to perform crossover (exchange binary information) based on the member's performance index. The better the performance index, the higher the probability that a member will be selected for crossover. Additionally, a mutation process is performed by randomly flipping some of the binary bits [170]. There are many ways to introduce randomness into a process like this. The idea is to have a higher probability of converging on the best solution, but still have a chance of exploring other parts of the design space.

GAs have been applied to many launch vehicle design and launch vehicle trajectory design problems [15, 35, 138].

2.3.3.2 Particle Swarm Optimization

Particle swarm optimization (PSO) is the inherently continuous version of GAs [170]. They are modeled after a flock of birds swarming around food. Each member of the population is represented by a position (the design variables) and a velocity. At every iteration in the algorithm, each member is updated in the following way:

$$X_k^l = X_{k-1}^l + v_{k-1}^l \delta t \tag{17}$$

Where X_k^l is the l member of the k population, and v_{k-1}^l is its corresponding velocity, which can be thought of as a search direction. The term δt is an arbitrary step size parameter. The velocity is updated each iteration as well based on an inertia parameter (basically how much is the new velocity dependent on the previous velocity), and two trust parameters that weight the best solution among all the members at the current iteration and the best solution of the member in question throughout all the iterations [170].

A PSO search is performed by initializing a population and updating the position of each member according to Equation 17. After evaluating all the performance indices, the velocity of each member is updated, and the algorithm is repeated [170].

A PSO algorithm was implemented in a study for Reusable Launch Vehicle trajectory optimization [39]. The study concluded that PSO algorithms work well even with small population sizes, which cuts down on computational costs.

2.3.3.3 Differential Evolution

Differential evolution (DE) is based off GAs but like PSO, is designed for continuous variables. It is arguably one of the most powerful stochastic global optimizers currently used [52]. Like GAs a population is seeded to start the optimization process. Each member is represented by a vector of design variables. To understand the algorithm some terminology is required. A current, or parent vector, is updated into

a trial vector (in the new generation) via a donor vector. The donor vector is created by using the parent vector and a weighting from two other vectors. Sometimes both other vectors are selected at random, while other times the best vector from the current generation is included with another random vector. The donor vector can be created on a vector by vector basis (i.e. for each donor vector, information from two other vectors are used), or on a variable by variable basis (i.e. each design variable in each donor vector uses information from two other vectors). In addition, the weighting between the two vectors can itself be a random number, again on a vector by vector or variable by variable basis. At this point the trial vector is created by randomly selecting the design variables from either the donor or parent vector. After all this the trail vector performance is compared to the parent vector performance and the best solution is kept. There are several variations of DE's and the parameters that control how the algorithm behaves can themselves be design variables in an optimization process [52].

DE algorithms have been applied to aerospace problems. Specifically, it was applied to the ascent launch trajectory optimization of a hypersonic vehicle, and was shown to outperform the equivalent NLP problem [67].

2.3.4 Global Methods Summary

There are several challenges associated with global searches. The most obvious is the computational requirements. Global methods are able to explore large design spaces, but the number of cases required can lead to infeasible run-times, even with state-of-the-art hardware. Another big challenge is termination criteria; i.e. when to stop the algorithm. There are several ways to terminate a global algorithm. The simplest is a hard limit on number of generations. More sophisticated methods involve looking at how much improvement has occurred over a certain amount of time, or how clustered the population is in the design space, or how much the current "best" point moves

vs how much the performance index changes. Because relevant global algorithms are stochastic, convergence (recall from Section 2.1.1.2 that convergence refers to finding the global optimum) is not guaranteed without an infinite number of cases [101]. Finding termination criteria that leads to good solutions without excessive run-time is challenging. Finally, because relevant global algorithms are stochastic, solving the same problem a second time will yield a different and possibly better result. For each problem it is important to determine how to best allocate computational resources, whether to run one search for a long time, or multiple shorter searches.

The advantage of global methods is that they are designed specifically to find the global optimum. In fact, they are heralded as "more likely than other methods to locate the global minimum" [46].

2.4 Selection of Optimization Technique

The previous sections have presented several different options for launch vehicle trajectory optimization. Meeting the goal of this thesis, from Section 1.3, requires data from a trajectory optimization method to generate a surrogate model. Figure 15 gives a breakdown of the trajectory optimization methods described in this chapter. Choosing from all these options leads to the first research question.

Research Question 1 - Which optimization technique should be used to generate the performance data?

Considering the depth of literature regarding trajectory optimization methods, this question can be answered from literature. Choosing a method from literature means a new method does not have to be developed and tested. Additionally, it is

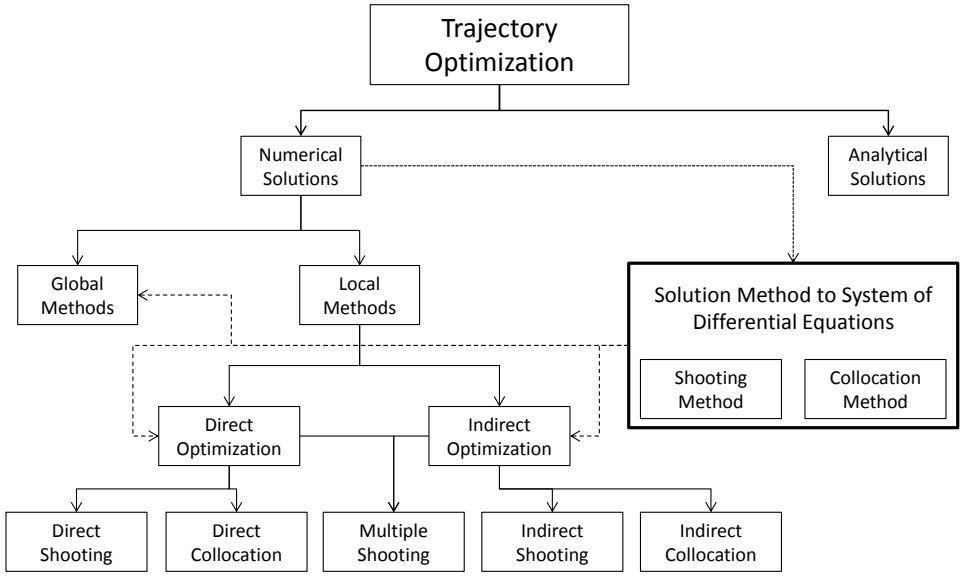


Figure 15: Launch vehicle trajectory optimization options

desirable to select a method that has been implemented in a general purpose trajectory optimization tool to reduce development time.

The development of a surrogate model will require on the order of $100's$ or $1000's$ of vehicles. In this context, automation is necessary. It is not reasonable to expect an analyst to manually set up trajectory optimization runs for 1000 vehicles in the time it takes to go through a conceptual design study. Furthermore, the method selected should be robust to minimize the number of failures. With this in mind, the available methods are tools are compared and contrasted. The discussion here is a summary of the conclusions from Section 2.3.

Indirect methods have been shown to not be robust, and therefore extremely difficult to automate. This poses a challenge for their use in this context. The other option is direct methods. Recall that global methods still require a way of representing the problem, so will still rely on either a direct or indirect approach. Direct methods are robust and easy to implement when compared to indirect methods and run relatively fast. The shortcoming of direct methods is that they do not always find the global optimum. Global methods are designed to find the global optimum,

Figure 16: Combined global and local optimization

but considering the number of vehicles to be evaluated (on the order of 1000's) quickly become infeasible [101]. With indirect method being intractable, direct methods will be used to formulate the problem. This means the control function $u(t)$ will be parametrically approximated, which has been shown to be a good option for launch vehicle trajectory optimization [19, 133, 173].

The challenge of not always finding the global optimum can be addressed by combining direct method local optimization with a global method. This strategy has been implemented in other domains as well as specifically to trajectory optimization [46, 75, 101]. A short and fast global search can be used to populate initial guesses for a local direct method. The combination of a global initialization and a direct optimization method meets of the needs of the trajectory analysis required here. It is computationally feasible, simple to automate, and yields the necessary performance data. What will result from this trajectory analysis is a set of performances from each of the direct method starting points. Because the answer is dependent on the initial guess, these performance values will not be the same. Recall that when using a direct method for trajectory optimization the final answer is dependent on the initial guess. Typically the best of the resulting performance values is reported, as illustrated in Figure 16. However, this can have adverse effects when the data is used to fit a surrogate model. A different method that leverages all the information from the resulting set of performance values will be presented in Chapter 3.

Since general purpose trajectory optimization tools using the direct method exist and are available, there is no need to develop a new optimizer. Two tools created

specifically for launch vehicles are POST and OTIS. Either of these tools can be employed. For this thesis, the author has chosen to use POST based on familiarity and availability. It has been shown that the outputs of POST and OTIS are similar [119], and the overall methodology presented here does not require one or the other, so POST is chosen. It is additionally beneficial that POST is an industry-standard tool that will give credibility to the optimized results.

The answer, then, to Research Question 1 is to use a global initialization of initial control parameter guesses for a direct method algorithm. The trajectory analysis tool POST will be used to optimize the trajectories and provide the corresponding performances.

2.4.1 Phase Discretization

Direct methods are a common way to solve launch vehicle OCPs in aerospace engineering [19, 69, 78, 99, 157, 176]. As discussed, direct methods are often preferred over indirect methods in many applications because they do not require the derivation of the necessary conditions [19] or an initial guess on the costates and have a larger convergence domain [20, 46, 157]. Specifically, the direct shooting method is commonly employed for launch vehicle ascent trajectory problems in conceptual design [17, 19, 176]. In a direct shooting method the optimal control function is approximated using a set of parameters in a process called transcription. The OCP is thereby converted into a NLP problem [78]. The NLP problem seeks to find the parametric solution that optimizes the cost function of interest. For launch vehicle OCPs, the parametric approximation can be accomplished with a small number of values representing a piecewise continuous or even piecewise linear function [17, 19, 63].

One of the challenges of employing this parametric method is that a mapping is required for the parametric approximation of the optimal control function. This

mapping, or control structure, defines how the parametric control replaces the optimal control function. In the transcribed representation of the OCP, each control parameter represents one degree of freedom. However, in the original OCP, the control is a function, representing infinite degrees of freedom. In the limit, an infinite number of parameters could be used to represent the control function. This is not a feasible solution so in practice each parameter is used to represent a specific section of the optimal control function. This results in a formulation where each parameter corresponds to a single degree of freedom in the NLP problem, but represents infinite degrees of freedom in the OCP. The control structure is required to define how these single degree-of-freedom control parameters will represent the optimal control function. Even though direct trajectory optimization is commonly used in industry, there is no systematic repeatable process that the authors are aware of to choose suitable control structures. Currently, control structures are developed *a priori* by trajectory analysts using experience and/or previous trajectories [4, 5, 162]. The optimized vehicle performance is sensitive to the control structure, so it is important to select suitable control structures. In the context of launch vehicle trajectory optimization, choosing the wrong control structure can result in inaccurate performance measures. If this occurs in conceptual design a sub-optimal vehicle may be selected, or an optimal vehicle rejected, for the next design phase. This can lead to expensive schedule slips and design changes. A traceable and repeatable method for generating control structures for trajectory optimization problems is needed for use in launch vehicle conceptual design.

2.4.2 Control Structure Refinement Methods

There have been many previous efforts related to refining what is referred to here as the control structure for OCPs. In the literature, this is generally referred to as mesh refinement instead of control structure refinement. The term mesh can refer

to several different aspects of the OCP, however. The number of points used in the numerical solution of the differential equations can be referred to as a mesh as well as the number of parameters used to approximate the control function [21, 82]. In addition, a very common method is to use some interpolating function to approximate either the state or the control (or both) parametrically using a structure also referred to as a mesh [51]. In this thesis, the only mesh considered is referring to the direct approximation of the control function using a set of parameters. Therefore, the term control structure is used to refer specifically to this mesh.

The discussion here will start with mesh refinement methods that can be applied for the numerical solution of differential equations. These methods, sometimes referred to as adaptive grid methods, are used in transcribed OCP's to find the best grid, or discretization, for numerically integrating the ODE's in the resulting NLP problem. Three general types of adaptive grid methods exist: h, p, and r refinement. In h-refinement extra nodes are added at strategic points to increase accuracy in critical areas. In p-refinement a different numerical method is used to solve the equations of motion in critical areas. Finally in r-refinement a fixed number of nodes are moved around to find the best distribution. Historically, grid methods have been applied to node distribution in the spatial domain [5]. In trajectory optimization, however, most node distribution is in the time domain. When using adaptive grid methods there is a tradeoff between computational effort and improvement of the solution [4]. If only solution accuracy was considered, an infinite number of grid points would be used, and the NLP problem would approach the optimal control problem. Figure 17 shows how run-time increases for a specific problem as the number of nodes increases. There can be seen a significant increase in computational time vs number of nodes, and as the number of nodes increases, the rate at which the CPU time increases as well, leading to what appears to be an exponential growth trend.

The effect of adaptive node placement vs a uniform on the control error is shown

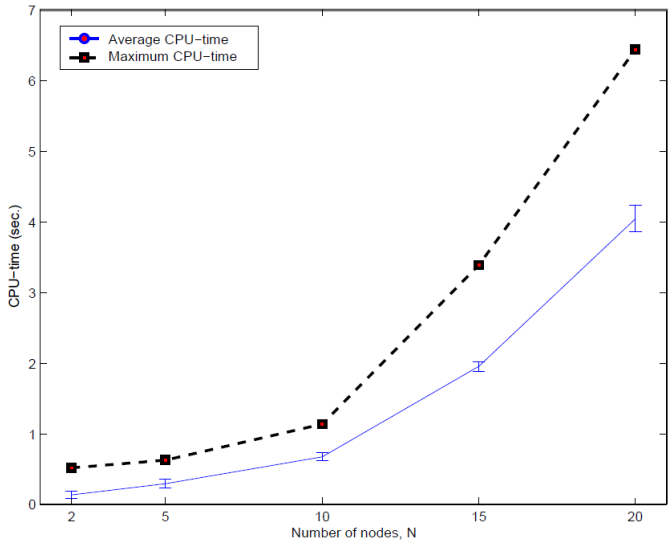


Figure 17: Computational cost vs number of nodes [5]

in Figure 18. The control error is plotted against the number of nodes used for adaptive and uniform methods. It is interesting to note that increasing the number of nodes does not always increase the accuracy of the result. The source noted that this was because a local optimizer was used, and global optimality was not guaranteed [170]. In theory, increasing the number of nodes will always decrease the error of the NLP solution when compared to the optimal control solution. A local algorithm may get bogged down with so many variables and be ineffective in finding a solution. It becomes a question, however, of computational effort, and a small improvement may not be worth the added expense.

These methods are effective in increasing accuracy, whether by inserting more points or shifting a set number of points, in local regions of the solution [82], but inherently rely on knowledge of the underlying differential equations. If a differential equation was found for the control, these methods could prove useful. In the application of this thesis, however, such a relationship is not available, and the solution method desired here cannot rely on knowledge of the underlying differential equation for the control.

Similar challenges exist with mesh refinement methods designed specifically for

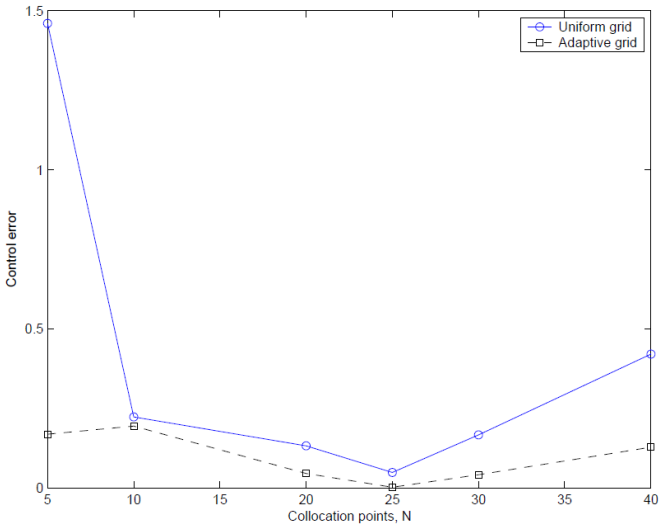


Figure 18: Control error for different grid methods [4]

OCPs. Betts et al. [20, 21] proposes a method for mesh refinement when the mesh refers to the points used in the numerical solution of the differential equations as well as the points used to parametrically approximate the control function. The measure of the discretization error, however, is based only on the state variables and relies entirely on the system of differential equations, meaning it does not take into account any information from the control parameters. It is assumed that control is optimal and that decreasing the error in the state variables will decrease the error in the control as well. The authors do provide examples to show that this approach is valid, but it requires that the control function be approximated at each time step used in the solution of the differential equations. The application in this thesis is the trajectory of a launch vehicle into orbit, which can be accurately modeled with a small number of parameters [19, 133, 173]. Implementing the method described in this paragraph would dramatically increase the complexity, increasing the number of control parameters form a handful (on the order of 10), to thousands.

Other mesh refinement solutions include using a density function for the location of the points [180]. The problem at this point becomes selecting the correct density function to distribute the points. This has been shown to be useful for some problems,

but it is not suited to the application at hand, where there are several distinct stages of flight. A density function would need to be designed specifically to take these stages into account. While this may provide a feasible solution for a single vehicle, if multiple vehicles are considered the density function would have to account for the differences in staging events for each vehicle.

Another common group of mesh refinement methods apply to the solution of OCPs using pseudospectral methods. These methods are a subset of the direct collocation method described in Section 2.3.1.3. The pseudospectral method uses a specific set of interpolating polynomials to approximate both the states and the controls between a set of grid points. Various mesh refinement methods applicable to pseudospectral solutions have been implemented [51, 68, 102, 103]. These methods can refine the mesh by inserting additional grid points or by increasing the degree (and therefore the flexibility) of the interpolation polynomial, or both. It is obvious that increasing the degree of the polynomial is not applicable to the solution method used in this thesis because single parameters, instead of polynomials, are used to approximate the control function. The addition of more grid points in these methods is based on information from either the states or from the polynomials used to approximate the control. As discussed earlier, applying a method such as this in this thesis would require a significant increase in the number of parameters used to approximate the control function.

In addition to the examples discussed here, there have been other efforts in determining how to place the nodes for numerical integration for the solution of ODE's [139], both for optimal control problems in general [40, 56, 92] and specifically for trajectory optimization problems [73, 137]. However, the challenge in this thesis deals with the node placement for the control parameters required in the transcription of the optimal control problem to an NLP problem. This node placement is currently not part of the optimization process. The main difference between node placement

for a set of ODE's and a control function is that the ODE's are known, while the control function is not. There have been efforts in spacecraft trajectory optimization to optimize the control structure as part of the overall optimization problem [46]. In this implementation, several phase structure variables are included as parameters in the optimization routine. No results were reported, however. A more detailed discussion of these methods is provided by Jain [82]. As far as the author can tell, there is no method that is applicable to the solution method implemented here, where the grid points for the numerical solution of the differential equations and the grid points for the control are independent and the control function is approximated using a small number of parameters. This leads to the second research question. The question is labeled 2.1 instead of 2 for reasons explained later in the document.

Research Question 2.1 - How should the control structure for a launch vehicle trajectory optimization problem be selected?

Research Question 1 dealt with how to evaluate the trajectories. Because of the wealth of literature regarding trajectory optimization, the question was answered using published work. The choice was to use a combination of global and local direct methods. The direct method requires a control structure that is specified as the problem is set up. Therefore a control discretization that applies to the vehicle being evaluated must be input. Unlike Research Question 1, the answer to this question is not found in the literature. Instead an experiment will be conducted to address it. In Section 2.6, a hypothesis will be developed based on the literature and a discussion of the problem and a method specifically tailored to the solution method selected will be presented. Before discussing this experiment, however, a relevant launch vehicle problem that captures the inherent challenges of launch vehicle trajectory

optimization is required as a case study. It is desirable that the data required to model this vehicle be publicly available. With that in mind, the following section will present the Delta IV Heavy launch vehicle and a relevant design space.

2.5 Delta IV Heavy

The Delta IV Heavy launch vehicle was developed as the largest variant in the Delta IV family of vehicles. It was first launched in 2004, even though other Delta IV variants had been in operation for over a decade [81]. The Delta IV vehicle has been modeled before in public academic studies [153, 161, 162]. In addition, there are several other sources for vehicle parameters that make it possible to model the vehicle accurately [24, 81, 169].

The Delta IV vehicles are all based on the Common Core Booster (CBC), a liquid oxygen (LOX) hydrogen stage powered by a single RS-68 Rocketdyne engine. The Delta IV family was designed in a modular fashion around the CBC to service different missions and payloads. This is achieved by strapping on up to four solid rocket motors for variations on the Delta IV Medium or by attaching three CBC's together for the Delta IV Heavy. The second stage is different for the Delta IV Medium and its variants and the Delta IV Heavy. In both cases, the upper stage is a LOX hydrogen stage powered by a single Pratt & Whitney RL10B-2 engine [81].

Table 2 gives the masses for the Delta IV Heavy. The core and boosters are three CBC's strapped together, and therefore their masses are the same. This gives a gross lift off mass of about 1.6 million lb. Table 3 shows the propulsion parameters for the Delta IV Heavy. This vehicle can be considered as a two-and-a-half stage vehicles. The first stage being split into a period when the core is burning with the two boosters and a period when the core alone is burning after the boosters have been staged. Taking into account the back pressure losses at sea level, lift-off thrust is over 1.9 million lb, giving the vehicle at thrust-to-weight of just under 1.2.

Table 2: Delta IV Heavy vehicle weights [81]

Vehicle Element	Weight (lb)	
Core Burnout	59000	
Core Propellant	440000	
Booster Burnout	59000	$\times 2$
Booster Propellant	440000	$\times 2$
Upper Stage Burnout	7700	
Upper Stage Propellant	60000	
Payload Fairing	6470	
Payload	63500	

Table 3: Delta IV Heavy vehicle propulsion parameters [81]

Propulsion Element	CBC	Second Stage
Engine	RS-68	RL10B-2
Vacuum Thrust (lb)	751000	24750
Vacuum ISP (s)	409	462.4
Exit Area (ft^2)	49.9	48.9

Modeling a launch vehicle trajectory problem requires a mission as well. The mission was selected from United Launch Alliance's Delta IV Launch Services User's Guide [169]. The target orbit is a typical circular LEO at an altitude of 400 km and an inclination of 28.7°. The mission launches from Kennedy Space Center (KSC) at a latitude of 28.5° and a longitude of 279.4°. The quoted performance capability for this orbit from the KSC launch site is about $63,500$ lb.

Aerodynamic data for the Delta IV Heavy is not publicly available. However, some form of aerodynamic data is necessary to simulate any ETO trajectory. The Air Force Research Laboratory (AFRL) has developed a semi-empirical design tool (Missile DATCOM) to calculate aerodynamic data for a wide variety of different vehicle configurations and flight conditions [171]. MDATCOM is intended as a preliminary design tool for missiles [23]. However, many launch vehicles, including the Delta IV, are similar in shape to missiles. Using non-dimensional aerodynamic coefficients allows scaling between smaller missiles and larger launch vehicles. MDATCOM has been used for launch vehicle aerodynamic calculations [153], and can be used to

estimate the aerodynamic data necessary to model the Delta IV.

MDATCOM takes as input a set of points that represent distance along an axis and the corresponding distance from that axis. The vehicle profile is represented by this set of points, and the profile is rotated around the axis to generate an axisymmetric representation of the launch vehicle. The Delta IV Heavy was modeled using Vehicle Sketch Pad (VSP), a NASA open source parametric geometry tool. Figure 19 shows the layout.

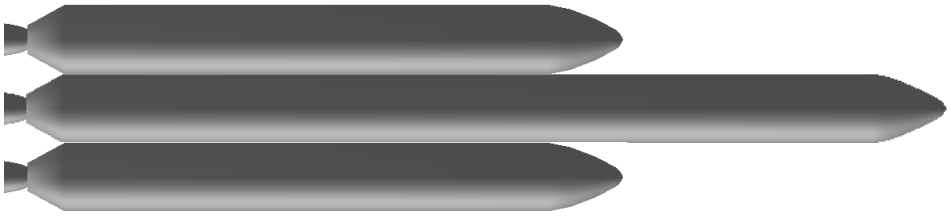
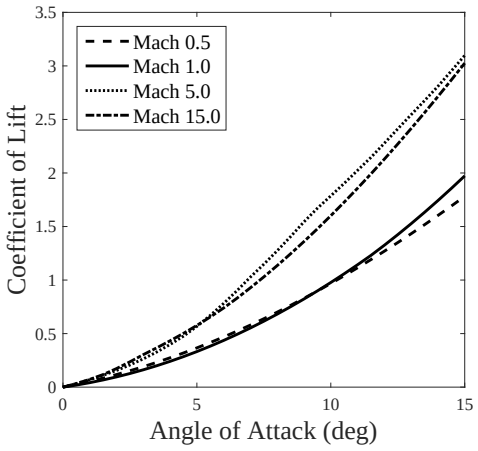


Figure 19: Delta IV Heavy model

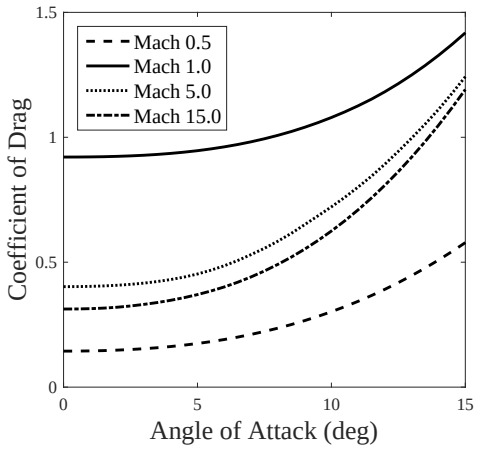
For this problem, two different configurations were modeled. The first is the configuration shown in Figure 19. This is how the vehicle is launched. Once the boosters are staged, the vehicle is simply the core and upper stage. Aerodynamic data is required for both these configurations.

MDATCOM outputs coefficients of lift and drag for each of the input flight conditions. The flight conditions of interest for this problem are angle of attacks between $-20°$ and $20°$ and Mach numbers between 0 and 15. The vehicle will experience Mach numbers higher than this during its trajectory, but only when it has reached an altitude where aerodynamic forces become negligible (less than 1 lb). The reference aerodynamic area used for the calculation of both configurations (with and without boosters) is the diameter of the core, 219 ft^2. Figures 20(a) and 20(b) show the aerodynamic data for the Delta IV Heavy model at relevant flight conditions with boosters. Figures 20(c) and 20(d) show the aerodynamic data for the core alone.

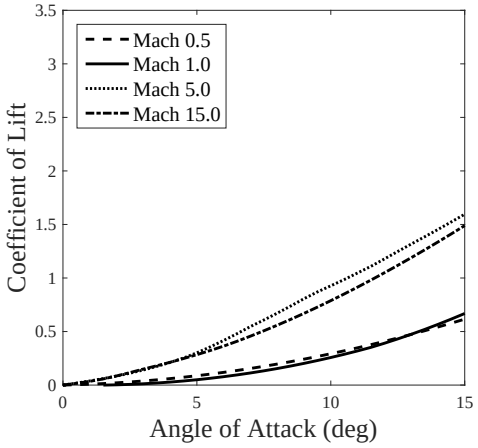
The concept of operations for the trajectory is based on the Delta IV Heavy sequence of events for a LEO mission [169]. Initially, the vehicle rises vertically



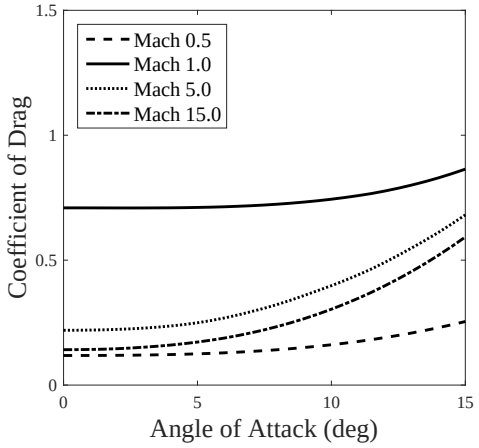
(a) C_L with boosters



(b) C_D with boosters



(c) C_L without boosters



(d) C_D without boosters

Figure 20: Aerodynamic coefficients for Delta IV Heavy using MDATCOM

for 2000 ft to clear the launch structure, after which a gravity turn is initialized. At this point the boosters are at 100% throttle while the core is throttled down to 54%. After the gravity turn, the vehicle flies at 0° angle of attack and sideslip angle through maximum dynamic pressure. The vehicle as allowed to fly at non-zero angle of attacks after the dynamic pressure has reached its peak, limited to 500 lb/ft^2, and been reduced to 20 lb/ft^2. The boosters burn the available propellant and are jettisoned when empty. The core is then throttled up to 100% and jettisoned after the propellant is used. Three seconds after the core is jettisoned the upper stage ignites. The payload fairing is jettisoned when the free molecular heating rate (FMHR), given in Equation 18, is equal to 0.1 Btu/ft^2s and decreasing [169]. From an aerodynamic perspective, a new stage results after the jettison of the fairing, as well as after the jettison of the core stage. The point in the trajectory where these events occur are at an altitude where the aerodynamic forces are considered negligible (< 1 lb) so a new set of aerodynamic coefficients is not required.

$$\begin{aligned} FMHR &= c \times q \times v_{rel} \\ &\text{where } c = 0.00128593 \end{aligned} \tag{18}$$

The trajectory is controlled using inertial pitch rates. The direct shooting method requires a finite number of pitch rates be used to achieve the final orbit. The number of pitch rates and where they are applied is traditionally determined by the trajectory analyst. The objective function is the final weight in the target orbit, which is to be maximized. The phases and their specific values of the trajectory are given in Table 4, where Time is measured in seconds, Alt (altitude) in feet, q (dynamic pressure) in psf, $wprp_i$ (i^{th} stage propellant) in lb, and Velocity (inertial) in ft/s. An important note regarding the "Control" column is that control here is used to refer to phases in the trajectory that the optimization process is in control of. In Phase 4, for example, there is guidance occurring to keep the angle of attack at zero, but the optimizer has no control over this, and therefore it is not considered a control. For this thesis, the

parameters of concern are the controls that the optimizer can change.

Table 4: Phase structure for trajectory of sample problem

Phase	Description	Start		End		Control
		Criteria	Value	Criteria	Value	
1	Initialization and vertical rise	Time	0	Alt	2000	None
2	Gravity turn	Alt	2000	q	150	Pitch rate
3	Reduce alpha	q	150	Time[1]	10	None
4	Throttle down core	Alt	12000	n/a	n/a	None
5	Max dynamic pressure	Time[1]	10	q	20	None
6	First stage guidance	q	20	$prop_1$	15000	Pitch rate
7	Throttle down boosters	$prop_1$	15000	$prop_1$	4000	Pitch rate
8	Booster burnout	$prop_1$	4000	$prop_1$	0	Pitch rate
9	Jettison boosters	$prop_1$	0	Time[2]	3	Pitch rate
10	Coast	Time[2]	3	Time[2]	4	Pitch rate
11	Throttle up core	Time[2]	4	Time[2]	9	Pitch rate
12	Core Burnout	Time[2]	9	$prop_2$	0	Pitch rate
13	Jettison core	$prop_2$	0	Time[3]	5	Pitch rate
14	Coast	Time[3]	5	Time[3]	8	Pitch rate
15	Upper stage guidance	Time[3]	8	Velocity	25548.8[4]	Pitch rate

2.5.1 Delta IV Heavy Trajectory

The baseline vehicle values were validated by simulating the ETO trajectory and comparing results. The orbit was selected as a typical LEO orbit: a circular orbit at an altitude of 400 km and inclination of 28.7°. The payload was determined using performance capability curves in the ULA Delta IV Launch Services User's Guide [169]. The payload for the orbit of interest is $63,500$ lb.

The vehicle and mission were simulated using POST [128] with seven inertial

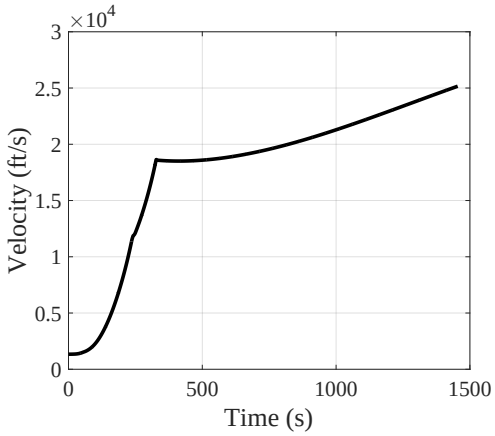
[1] Time here is measured from the beginning of Phase 3
[2] Time here is measured from booster burnout
[3] Time here is measured from core burnout
[4] The velocity required for circular orbit

pitch rates. The baseline POST input file is shown in Appendix B. The simulation results agree very well with the reported vehicle values. The Delta IV Heavy, as described in Tables 2 and 3, is quoted as being able to put $63,500$ lb of payload into the orbit described above. The simulation results achieve the same goal with 117 lb of propellant left in the upper stage. In terms of upper stage propellant, this is about a 0.2% error. Given that the aerodynamics of the vehicle were obtained using MDATCOM, a low fidelity aerodynamics software, this error is remarkably good.

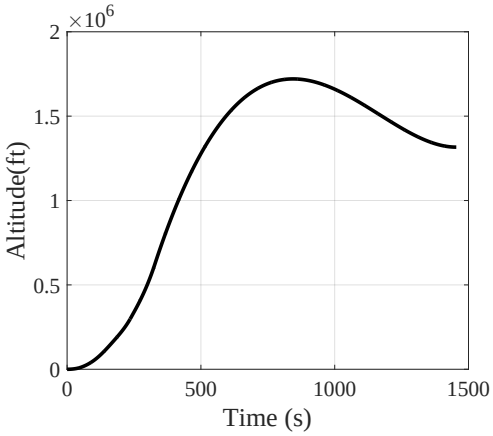
It should be noted that the Delta IV Heavy is being used here as an example problem. Indeed, any launch vehicle could be used, existing or not. The Delta IV Heavy was chosen based on availability of data and relative simplicity of the launch trajectory. If some parameters are different, or the trajectory simulation does not quite represent the actual vehicle flight, this does not invalidate the method being presented and tested here. In a launch vehicle design program, the most current definition of the vehicle and flight assumptions should be used if this methodology is applied.

The baseline trajectory is represented in a series of plots in Figure 21. Figure 21(a) shows the inertial velocity throughout the trajectory. The both staging events, booster and core, can be seen in the plot, with the booster staging occurring around 250 s and the core staging around 350 s. The second staging event is much more pronounced.

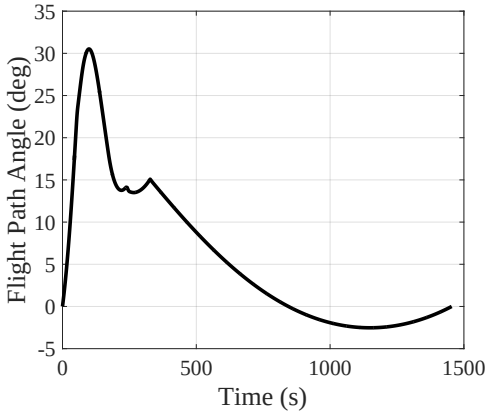
Figure 21(b) shows the altitude of the vehicle throughout the trajectory. The staging events cannot be clearly seen in this plot. This is because the vehicle will continue to ascend even if thrust levels change, as occurs during staging. On the velocity plot thrust changes due to staging will be seen clearly because the rate of change of velocity will be directly affected. The vehicle actually reaches a higher altitude than the target orbit before descending and inserting into the orbit. This is known as a lofted trajectory. Indeed sometimes it is more efficient to overshoot the



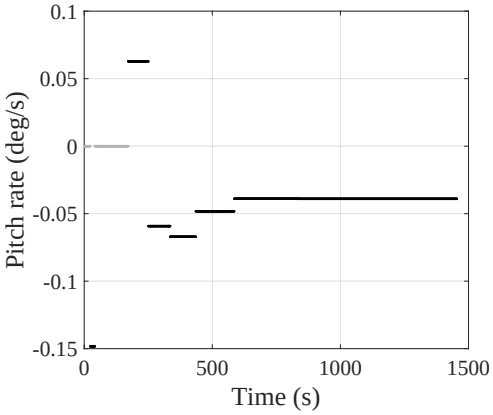
(a) Inertial Velocity



(b) Altitude



(c) Flight Path Angle



(d) Inertial Pitch Rate

Figure 21: Baseline trajectory for Delta IV Heavy

altitude and then descend as velocity is increased than it is to insert directly, especially with low thrust upper stages. This result has been seen before with vehicles inserting into LEO [48].

Figure 21(c) shows the flight path angle for the trajectory. Flight path angle starts at 0° because of the velocity from the rotation of Earth, which is tangential to the surface of Earth. As the vehicle gains vertical velocity the flight path angle increases. In this case, both staging events are clearly seen, as well as the initial gravity turn. The gravity turn is the initial smooth bump, taking the flight path angle up to about 30°. The booster staging is seen as the first small spike and the core staging is the second spike. The flight path angle does go below 0° around 800 s, which corresponds to the maximum altitude in Figure 21(b). After this point, the vehicle is traveling downward until it reaches the desired altitude, at which point the flight path angle is 0° and the vehicle is in a circular orbit.

Figure 21(d) shows the control variables for the trajectory. The gray lines at $0°/s$ represent phases of the trajectory that the optimizer is not in control of the vehicle. These times include the initial vertical rise (Phase 1 in Table 4) and the zero-alpha phase during maximum dynamic pressure (Phases 3-5 in Table 4). During these phases, the vehicle is guided to maintain a vertical rise or 0° angle of attack, so the optimizer cannot influence the flight. The darker lines represent the pitch rates that the optimizer does control to maximize the final weight in orbit. The final pitch rate, from about 600 s on is actually 2 pitch rates. The difference between these two controls is very small numerically. However, this difference is what controls whether or not the vehicle makes it to the final orbit, i.e. reaching the appropriate velocity, flight path angle, and altitude at the same time. Even small changes in the control variables can make large differences in the final trajectory.

2.5.2 Delta IV Heavy Design Space

The Delta IV Heavy is already a mature launch vehicle with flight history. For this thesis it is being used as an example problem. Assuming the vehicle was in the conceptual design phase, a design space for candidate vehicles would exist. Following is presented a design space that could be used for the Delta IV Heavy. This design space will be represented by a set of parameters defining a vehicle and a range on each of the parameters.

Tables 2 and 3 describe the baseline vehicle. The burnout and propellant masses for the boosters, core, and upper stage, along with the payload and fairing masses fully describe the weights of the system. If the core and boosters are not the same system, this results for 8 variables. The propulsion systems can be accounted for using thrust and ISP as variables. This adds 6 variables, 2 each for the core, booster, and upper stage. Finally, a path constraint is added to the system to limit the maximum aerodynamic loads. This is done by limiting the maximum dynamic pressure.

Summarizing, the design variables for a vehicle include core burnout mass, core propellant mass, booster burnout mass, booster propellant mass, upper stage burnout mass, upper stage propellant mass, payload mass, fairing mass, core thrust, core ISP, booster thrust, booster ISP, upper stage thrust, upper stage ISP, and maximum dynamic pressure.

If ranges are input on these variables as is, there exists the possibility of defining physically infeasible vehicles. For example, a high value for all the masses coupled with low values for the core and booster thrusts would define a vehicle that does not generate enough thrust at lift off and therefore will not fly. To avoid this issue, a different parameterization can be chosen to define the vehicles. The goal is to use the same amount of variables, but allow for a parameterization that will more likely yield feasible vehicles.

A good example of this is using thrust-to-weight for the upper stage. If the

total upper stage weight is defined, and a feasible thrust-to-weight is used, the thrust weight can be calculated. In either case two variables are used and the transformation is one-to-one and invertible. Some definitions for parameters like this are given in the equations below.

$$\text{Thrust to Weight (TW)} = \frac{Thrust}{Weight} \tag{19}$$

$$\text{Propellant Mass Fraction (PMF)} = \frac{\text{propellant mass}}{\text{total mass}} \tag{20}$$

$$\text{Inert Mass Fraction (IMF)} = \frac{\text{burnout mass}}{\text{total mass}} \tag{21}$$

Usually inert mass fraction is defined with inert mass in the numerator, but for the purposes of this thesis burnout mass is used. Additionally, the total mass refers to the upper stage vehicle, so it does not include mass of the payload fairing or the payload.

Using these equations, the following variables can be used instead of the ones listed above: vehicle thrust-to-weight, upper stage thrust-to-weight, upper stage IMF, booster PMF, core PMF, upper stage burnout mass, core burnout mass, core ISP, booster ISP, upper stage ISP, payload mass, fairing mass, and maximum dynamic pressure. This is a total of 13 variables. There are two missing. These two can be defined as the thrust and propellant mass ratios between the core and the combined boosters, shown in Equations 22 and 23.

$$\text{Thrust Ratio} = \frac{\text{combined booster thrust}}{\text{core thrust}} \tag{22}$$

$$\text{Propellant Mass Ratio} = \frac{\text{combined booster propellant mass}}{\text{core propellant}} \tag{23}$$

This allows for the core and boosters to be appropriate in relative size. For the Delta IV Heavy vehicle both these ratios would equal 2, as the boosters and core are all

Table 5: Design variable ranges for experiments

Variable	Abbreviation	Dimension	Baseline	Max	Min
Vehicle TW	G_TW	N/A	1.18	1.35	1.16
Upper Stage TW	US_TW	N/A	0.19	0.5	0.17
Upper Stage IMF	US_IMF	N/A	0.11	0.12	0.05
Core/booster ISP	CB_ISP	s	409	450	400

CBCs. The transformation between these two design spaces is shown in Appendix C.

Throughout this document, a set of research questions will be addressed via experimentation. This section describes a design space that applies to the entire vehicle. However, for this thesis, the scope of the problem will be reduced to consider 4 design variables.

The variables chosen are thrust-to-weights for the vehicle and upper stage, upper stage inert mass fraction, and the ISP for the core and booster. Previously, the core and booster ISP was separated into two variables, but for the experiments one variable will be used, and the core and boosters will have the same ISP values. Table 5 gives the variables and ranges used for the experiments.

2.6 Control Structure Experiments

Since Research Question 1 has been answered, the first experiment corresponds to Research Question 2.1, stated in Section 2.4.2. This is referring to the control structure used in the direct optimization approach. The direct method was the only method available for automation, and so it was required. Given that a direct method is used, a control structure must be selected. For the baseline trajectory in Section 2.5.1 a control structure was chosen based on an example in the literature [161]. However, this control structure was chosen *a priori* by an analyst. The following section will develop a method for finding a control structure in a traceable and repeatable manner.

It should be clarified here that the thrust of this experiment is to address how the control parameters, i.e. the pitch rates in this case, are applied to control the flight of the vehicle. Later on in the document, specifically in Section 3.4, the challenge

of estimating the performance of a vehicle by optimizing the value of the control parameters.

2.6.1 Method for Generating the Control Structure

A direct method trajectory optimizer uses a set of control parameters to approximate the control function. The number of parameters and how they are applied is referred to in this thesis as a control structure. A method is developed here for inserting additional degrees of freedom into an existing control structure in a way that maximizes the benefit to the NLP solution given the insertion of a single variable. The method is designed to generate control structures for the transcribed OCP that enable the optimizer to find improved costs. The method applies to OCPs solved using a direct shooting method and is designed to help solve an ETO launch vehicle trajectory problem. The results show that the method is effective in predicting which parameter insertion leads to the greatest cost improvement. The method can be implemented in an iterative process until the optimal insertion of an additional parameter results in inconsequential improvements. The method will be explained and demonstrated on two example problems and then implemented to answer Research Question 2.1.

A short review of the relevant information from previous sections is included here. The general OCP can be mathematically represented as finding the optimal control u^* that minimizes the cost function

$$J = \phi(t_f, x_f) + \int_{t_0}^{t_f} L(t, x, u)dt \tag{24}$$

subject to:

$$\begin{aligned} \dot{x} &= f(x, u, t) && \text{Dynamic constraints} \\ x(t_0) &= x_0 && \text{Initial conditions} \\ \Psi(t_f, x_f) &= 0 && \text{Terminal constraints} \\ S(x) &\leq 0 && \text{State inequality constraints} \end{aligned} \tag{25}$$

Here, J is the total cost and $\phi(t_f, x_f)$ and $L(t, x, u)$ are the terminal and running costs respectively. This is a common formulation for OCPs and has been used previously by Longuski [104] and Kirk [91]. There are two general approaches for solving this problem: indirect and direct. This thesis applies a specific formulation of the direct method known as control parameterization or direct shooting. In this formulation the state variables are propagated using a standard numerical integrator, such as Runge Kutta methods, and the optimal control function is approximated using a set of parameters [17]. This results in a NLP problem, shown in Equations 26 and 27. This approach is very common in industry for launch vehicle trajectory optimization in conceptual design because it is simple to use, relatively robust compared to other methods available, and lends itself to the ETO problem [19, 176].

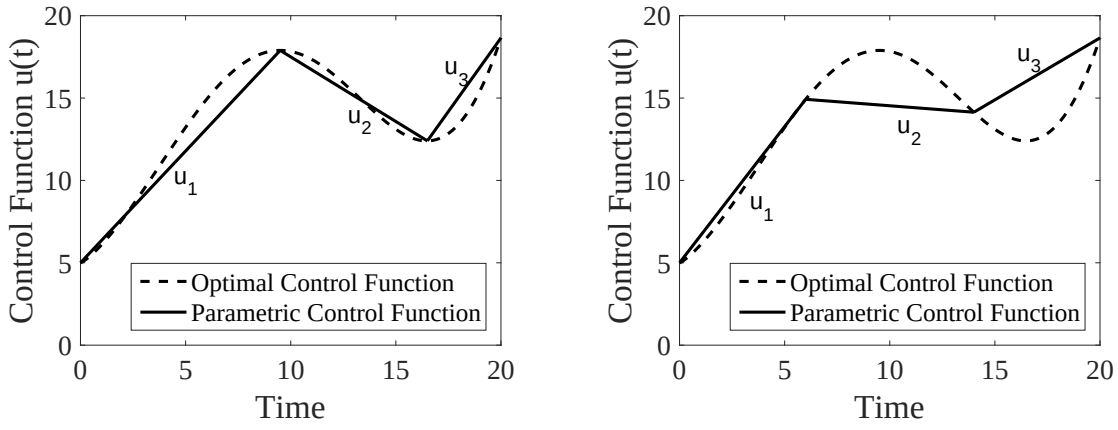
$$\text{Minimize: } F(\vec{x}) \tag{26}$$

subject to:

$$\begin{aligned} &g_j(\vec{x}) \leq 0 && j = 1, m && \text{inequality constraints} \\ &h_k(\vec{x}) = 0 && k = 1, l && \text{equality constraints} \\ &x_i^l \leq x_i \leq x_i^u && i = 1, n && \text{side constraints} \end{aligned} \tag{27}$$

Here, $\boldsymbol{X}$ is a vector X_1 through X_n, and there are m inequality constraints and l equality constraints.

In this formulation of the problem, the elements of the design vector $\boldsymbol{X}$ represent control inputs, denoted by u_i. When these control inputs are coupled with a control structure they make up a control function $u(t)$. This control structure is required to map the elements of the design vector $\boldsymbol{X}$ to the optimal control function $u(t)$. An example of this mapping, which is usually based on time, is seen in Equation 28. The times t_1-t_k define when the control inputs u_i are applied and will be referred to as a control structure ($\boldsymbol{U}$) in this thesis. The time values used to generate the control structure directly affect how well the control can approximate the control function. Figure 22 shows two different time-based parametric approximations of the same



(a) Example A

(b) Example B

Figure 22: Example control function discretization

notional control function $u(t)$. The approximations both use three parameters, u_1 - u_3, but these are applied using different control structures. In Figure 22(a) the function is relatively well approximated, while in Figure 22(b) the peak and trough of the control function is not captured. This illustrates that the control structure selection plays an important role in the ability of the parametric solution to approximate the optimal control function.

$$X_i = u_i, \quad t_i < t \leq t_{i+1} \quad \text{for} \quad i = 1, k-1 \quad \text{and} \quad t_0 < t_1 < ... < t_k \tag{28}$$

A control structure can be updated by inserting additional control parameters. In Figure 22(b), for example, an extra control variable may be added between Time 5 and Time 15 to capture the peak. The decision of where to add control variables is not this simple when the optimal control function is unknown and the wrong decision can lead to inaccurate cost measures. In launch vehicle conceptual design it may be necessary to rapidly and accurately solve many trajectory optimization problems without known control structures. The right performance measures are required to make decisions regarding potential launch vehicles. The automated method here is designed to find control structures for these problems.

The method presented here is based on the same principles that many NLP solvers

use. NLP solvers are designed to operate on the design variables $\boldsymbol{X}$ to minimize (or maximize) the cost function $F(\boldsymbol{X})$. A common approach is for the NLP solver to calculate the gradient of the cost with respect to the variables X_i, $\nabla F(\boldsymbol{X}) = [\partial F/\partial X_1, \partial F/\partial X_2, ...\partial F/\partial X_n]$, $\boldsymbol{X} \in \Re^n$, and then modify each X_i accordingly until a convergence criterion is reached. For transcribed OCPs, the elements of $\boldsymbol{X}$ are control parameters u_i, so the design variables $\boldsymbol{X}$ are equal to a vector $\vec{u}$. This vector $\vec{u}$ represents a control function $u(t)$ when coupled with a control structure. For any given formulation of the NLP problem, each control input u_i could be replaced by any number of parameters. For example, u_2 in Figure 22(b) could be replaced by u_{2a} and u_{2b}. This would increase the total number of elements in $\vec{u}$ by one, meaning the NLP solver can leverage the additional degree of freedom and modify u_{2a} and u_{2b} independently. If the partial derivatives with respect to u_{2a} and u_{2b}, given by $\partial F/\partial u_{2a}$ and $\partial F/\partial u_{2b}$, are unequal, the optimizer will modify them differently. The numerical difference between the derivatives of two replacement parameters (u_{2a} and u_{2b} in this example) serves as a relative measure of how much the function F would improve with the introduction of the extra degree of freedom. No difference in the derivatives indicates that the optimizer would modify the two replacement parameters in the same manner, and therefore the additional degree of freedom is not useful. On the other hand, a large difference in the derivatives indicates that the optimizer would modify the two replacement parameters differently, and therefore the problem would benefit from the additional degree of freedom. The derivative difference is a relative measure because it indicates which parameter replacement will result in the most improvement, but does not indicate how much improvement will result.

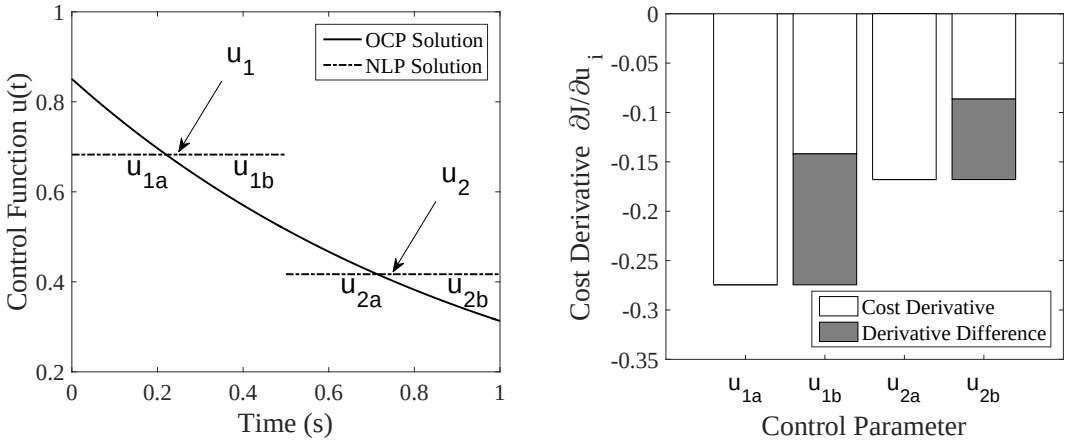
An example of parameter replacement and the derivative difference is illustrated in Figure 23. The solution to an OCP, which will be considered in depth in Section 2.6.1.1, is overlaid with an NLP solution in Figure 23(a). The NLP solution shows how a control vector $\vec{u}$ with two elements, u_1 and u_2, each applied for 0.5 s, can be used

to approximate the optimal control function. The accuracy of the approximation in terms of the final cost will be discussed in Section 2.6.1.1. Each element of $\vec{u}$ represented in Figure 23(a) can be split into two equivalent elements for a total of four control input parameters, labeled u_{1a}, u_{1b}, u_{2a} and u_{2b}. The partial derivatives of the cost with respect to these four parameters are shown in the bar graph in Figure 23(b). The difference between the partial derivatives of the cost with respect to u_{1a} and u_{1b} is illustrated by the gray bar in the u_{1b} column. The gray column in the u_{2b} column represents the derivative difference between u_{2a} and u_{2b}. The fact that the derivative difference is non-zero in both cases indicates that a better solution would be found by replacing both u_1 and u_2 with two parameters each. The larger derivative difference corresponding between u_1 and u_2 indicates that a greater benefit would result from replacing u_1 with u_{1a} and u_{1b} than by replacing the u_2 with u_{2a} and u_{2b}. Results confirming this will be shown in Section 2.6.1.1. A visual inspection of the optimal control function in Figure 23 reveals that the slope has greater average magnitude from $0 - 0.5$ s than from $0.5 - 1$ s. This means, on average, the u_1 is farther from the optimal control than the u_2. Therefore replacing the u_1 with u_{1a} and u_{1b} yields a better cost. It may seem trivial to decide which parameter to replace when the optimal control function is plotted in the same figure, but in practice, the optimal control function is unknown.

This leads to Hypothesis 2.1.

Hypothesis 2.1 - If a method based on the derivative difference is used to iteratively generate control structures, then that method will find control structures that maximize improvement at each iteration.

The method proposed is to use the derivative difference measure to iteratively



(a) Control History (b) Derivative Difference

Figure 23: Example NLP solution and corresponding derivative difference information

introduce additional degrees of freedom in the form of elements of the control vector $\vec{u}$ in a manner that maximizes the realizable cost improvement. An initial solution to the NLP problem is required to start. The derivative difference measure can be calculated for each element in $\vec{u}$. This measure can be found using finite differencing without any need for re-solving the NLP problem. The control input u_i with the highest derivative difference can then be replaced with two control parameters and the NLP problem re-solved. The process can be repeated indefinitely as degrees of freedom are added to better approximate the optimal control function. Figure 24 shows a flowchart of the proposed method. Steps $3-5$ represent the new contributions that will presented and demonstrated. Traditionally, Steps 1 and 2 would be carried out and either the control structure would be modified by an analyst or the user would proceed to Step 7 and the solution would be used. Step 6, the exit criteria diamond, represents some termination condition determined by the needs of the specific problem. Termination criteria could include a maximum number of elements in $\vec{u}$, computational expense, relative rate of solution improvement, etc. For the remainder of this thesis, the process illustrated in Figure 24 will be termed Optimal

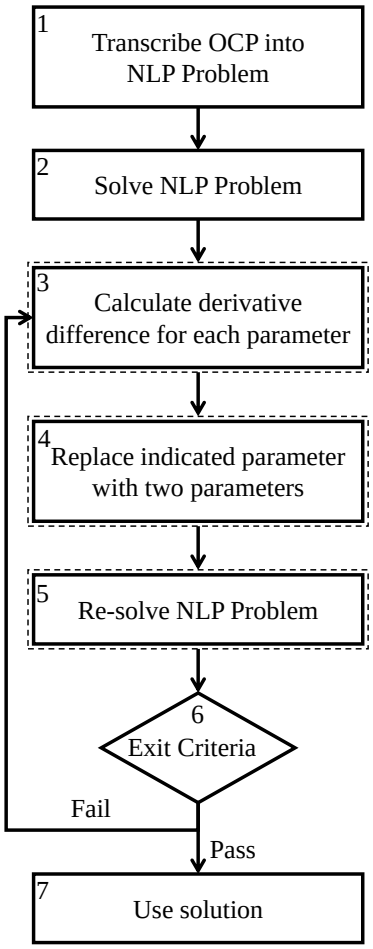


Figure 24: Steps for the OEPI method

Equivalent Parameter Insertion (OEPI) method. The term "equivalent" is included in the method name because only equivalent parameters are being considered. There are an infinite number of ways to replace a single element u_i with two, but for this thesis, the elements will be split equivalently. Other replacement methods is a topic for future work.

The OEPI method is demonstrated experimentally in the following sections. Two example problems have been chosen. The first is a very simple one-dimensional OCP with an analytical solution. The OEPI method will be applied to several different initial parametric solutions to demonstrate and experimentally verify the method. The second problem is a lunar launch problem, again with an analytical solution.

The OEPI method is applied and experimentally verified. Finally, the OEPI method is applied to the ETO trajectory optimization for the Delta IV Heavy using standard conceptual design methods [176].

2.6.1.1 Simple Optimal Control Problem

The OEPI method is applied to the following notional 1-D simple OCP.

$$\min_{u(t)} \int_{t_0}^{t_f} u^2(t)dt \tag{29}$$

subject to the dynamic model:

$$\dot{x} = x + u(t) \tag{30}$$

with boundary conditions:

$$x(0) = 0 \qquad x(1) = 1 \tag{31}$$

This problem will be solved analytically first and then using direct shooting to illustrate the OEPI method.

Analytical Solution The Hamiltonian can be computed as

$$H = u^2 + \lambda(x + u) \tag{32}$$

where λ is the Lagrange multiplier. The Euler-Lagrange equation can then be solved, yielding the following relationships:

$$\dot{\lambda}(t) = -\frac{\partial H}{\partial x} = -\lambda \tag{33}$$

$$\frac{\partial H}{\partial u} = 0 = 2u + \lambda \quad \Rightarrow \quad u = -\lambda/2 \tag{34}$$

Equation 33 can be solved for λ.

$$\lambda = Ce^{-t} \tag{35}$$

This leads to the optimal control, denoted by $u^*(t)$.

$$u^*(t) = -\frac{C}{2}e^{-t} \tag{36}$$

The only remaining step is to solve for the constant C using the necessary conditions for optimality. This can be done by using the fact that the Hamiltonian is time invariant for this system and therefore $H(t_0) = H(t_f)$. This can be combined with the initial and final conditions in Equation 31 to solve for C, which is given Equation 37.

$$C = \frac{4e^{-1}}{e^{-2} - 1} \approx -1.70184 \tag{37}$$

Substituting this constant into the control function

$$u^* = -\frac{2e^{-1}}{e^{-2} - 1} e^{-t} \tag{38}$$

The cost function can then be directly evaluated from Equation 29.

$$\min_u \int_{t_0}^{t_f} u^2 dt = \int_0^1 \frac{4e^{-2}}{(e^{-2} - 1)^2} e^{-2t} dt = \frac{4e^{-2}}{(e^{-2} - 1)^2} \frac{1}{2}(1 - e^{-2}) \approx 0.313035 \tag{39}$$

The optimal state and control solution for this problem are plotted in Figures 26(a) and 26(b).

Numerical Solution using Direct Shooting An alternate way to solve this problem is to use the direct shooting method. Direct shooting is implemented here to illustrate the OEPI method even though the analytical solution is already known. The OCP being solved is posed as an NLP in the following equations.

$$\text{Minimize} : F(\boldsymbol{X}) = \int_{t_0}^{t_f} u^2(t) dt \tag{40}$$

The optimization function $F(\boldsymbol{X})$, given in Equation 40, is calculated using the dynamic model and boundary conditions in Equations 30 and 31. The design variable vector $\boldsymbol{X}$ is chosen to be two control parameters, shown in Equation 41. This can be combined with a control structure $\boldsymbol{U}$, shown in Equation 42, to generate a control function $u(t)$.

$$\boldsymbol{X} = \vec{u} = \begin{bmatrix} u_1 \\ \\ u_2 \end{bmatrix} \tag{41}$$

$$\boldsymbol{U} = \begin{cases} u_1, & 0 \leq t < 0.5 \\ u_2, & 0.5 \leq t < 1 \end{cases} \tag{42}$$

There are an infinite number of elements and control structures that could be used here. This control contains two equivalent elements: u_1 being applied from 0 to 0.5 seconds and u_2 being applied from 0.5 to 1 second. A standard unconstrained optimizer, such as the *fminunc* function in MATLAB [109], can be used to solve this problem by adding a terminal cost to represent the second boundary condition in Equation 31 in an overall evaluation criterion (OEC) as seen in Equation 43, where x_{target} is the second boundary condition in Equation 31. This OEC is used instead of the function in Equation 40. Note that when the boundary conditions are met, Equations 40 and 43 are equivalent.

$$J = \mathbf{OEC} = \left[\left(x(t_f) - x_{target}\right) C\right]^2 + \int_{t_0}^{t_f} u^2(t) dt \tag{43}$$

Here, C is some weighting coefficient for the terminal cost. For this problem, C was set to 10^4. This ensured that the terminal conditions were met.

A generic solution process is shown in Figure 25 with this specific problem as an example. The dynamic model is the equations of motion and boundary conditions, given in Equations 30 and 31, and the control structure is shown in Equation 42. The initial guess is simply numerical values for the controls to initialize the problem. The optimizer then modifies the values to minimize the cost until a convergence criteria is met. This criteria could include number of iterations, relative improvement of the cost, relative change in the design variables, etc. The solution that the optimizer arrives at depends on the initial guess given. For this problem, and the rest of the transcribed problems in this section, different initial guesses are run to ensure an accurate representation for the best parametric solution possible is reported.

A solution for this problem is given by $u_1 = 0.683$ and $u_2 = 0.417$. The resulting trajectory is compared to the optimal trajectory in Figure 26(a). Figure 26(b) shows

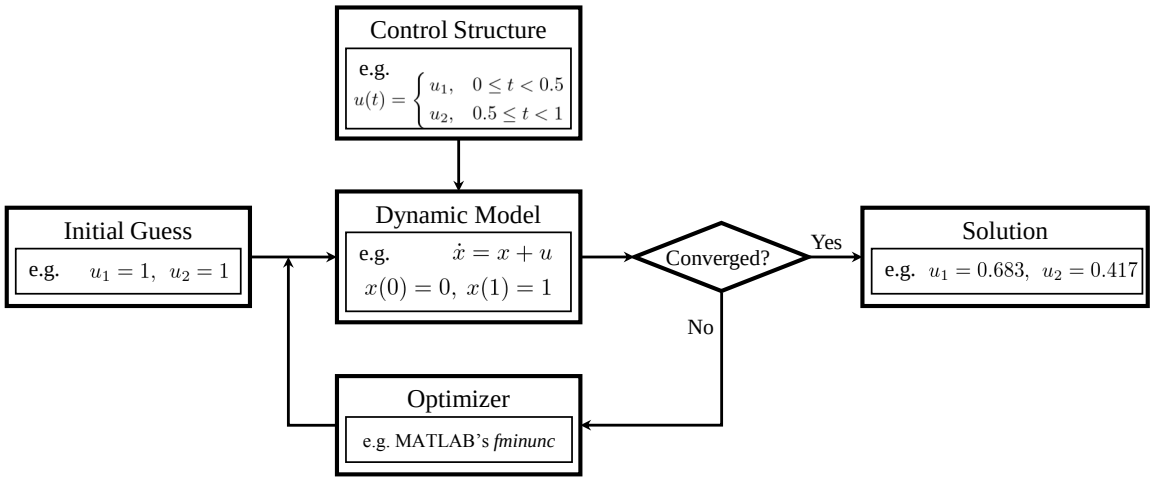


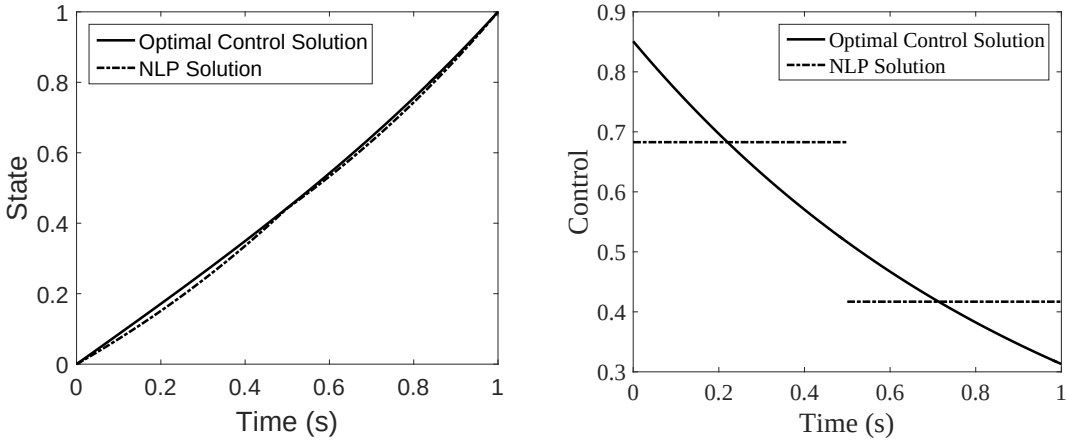
Figure 25: NLP solution method

the optimal and transcribed control. The difference in the problem formulation is seen clearly when the two controls are compared. The optimal control is a function of time, whereas the transcribed control is two control values, each applied for a specified set of time. It is interesting to note that the state histories for this example do not show significant variation even though the controls are drastically different.

This solution to the transcribed OCP leads to a cost of 0.320 (units are not included for this notional cost). This is higher than the cost of 0.3130 when the analytical optimal control is used. It should be noted that there are several parametric control solutions that will yield the same (or very similar) cost for the transcribed problem.

Application of the OEPI method In the previous paragraphs the problem described in Equations 29 - 31 was transcribed and solved using the control in Equations 41 and 42, which contained two elements. Now the OEPI method will be applied to find a lower cost. The steps in Figure 24 are used as a guide for the process.

Steps 1 and 2 are to transcribe and solve the NLP problem, which was done in the previous section. The solution using the control structure given in Equation 42



(a) State history (b) Control history

Figure 26: Comparison of optimal and transcribed problem solutions for simple problem

was found to be

$$u(t) = \begin{cases} u_1 = 0.683, & 0 \leq t < 0.5 \\ u_2 = 0.417, & 0.5 \leq t < 1 \end{cases} \tag{44}$$

This solution resulted in a cost of 0.320.

Step 3 is to calculate the derivative difference for each parameter. This is enabled by splitting each control parameter into two without changing the values. The replacement is shown in Equation 45. At this point the control $u'(t)$ still represents the same solution as the control $u(t)$, and the state and control histories are identical. However, that the number of elements and the control structures differ.

$$u(t) = \begin{cases} u_1 = 0.683, & 0 \leq t < 0.5 \\ u_2 = 0.417, & 0.5 \leq t < 1 \end{cases} \rightarrow u'(t) = \begin{cases} u_{1a} = 0.683, & 0 \leq t < 0.25 \\ u_{1b} = 0.683, & 0.25 \leq t < 0.5 \\ u_{2a} = 0.417, & 0.5 \leq t < 0.75 \\ u_{2b} = 0.417, & 0.75 \leq t < 1 \end{cases} \tag{45}$$

A simple finite differencing method can be used to calculate the derivatives of the cost with respect to each of the elements in $u'(t)$. The derivative difference measures corresponding to each element in $u(t)$ are then used to determine where to insert

an additional element to generate the new control structure. It is important to note that at this step the NLP problem is not solved using $u'(t)$. This control structure is simply used to enable the calculation of the derivative differences. This calculation is done without re-solving the problem. For this example the derivative of the cost with respect to the first element of $u'(t)$, denoted by $\partial J/\partial u'_1$, was -0.275. This was compared to $\partial J/\partial u'_2 = -0.142$ and the derivative difference, $\partial J/\partial u'_1 - \partial J/\partial u'_2$, was -0.133. The derivative difference for the second pair, $\partial J/\partial u'_3 - \partial J/\partial u'_4$, was -0.0815.

Step 4 in the process is to replace the parameter indicated by the derivative difference measure with two. For this method, only the absolute value of the difference is of interest. In this case, the derivative difference indicates that a better solution would be found if u_1 was replaced by u_{1a} and u_{1b} instead of replacing u_2 with u_{2a} and u_{2b} (refer to Equation 45). The control structures resulting from both these options are shown in Equation 46. The derivative difference indicates that $\boldsymbol{U}_a$ should be used.

$$\boldsymbol{U}_a = \begin{cases} u_1, & 0 \leq t < 0.25 \\ u_2, & 0.25 \leq t < 0.5 \\ u_3, & 0.5 \leq t < 1 \end{cases} \quad \text{and} \quad \boldsymbol{U}_b = \begin{cases} u_1, & 0 \leq t < 0.5 \\ u_2, & 0.5 \leq t < 0.75 \\ u_3, & 0.75 \leq t < 1 \end{cases} \tag{46}$$

Step 5 is to re-solve the NLP problem with the resulting control structure, as seen in $\boldsymbol{U}_a$ in Equation 46. This results in a cost of 0.317, which is an improvement compared to the cost of 0.320 when the control structure in Equation 42 is used. Although not part of the OEPI method, in this study this result will also be compared to re-solving the problem using the control structure $\boldsymbol{U}_b$ in Equation 46. Results from this comparison will be discussed later.

The final steps in the process shown in Figure 24 are based on the specific application. The user must decide if the solution has met the exit criteria and then put the solution to use. Specific criteria are not addressed here; that decision is left to be determined for each specific application. If the exit criteria has not been met, the Steps $3-5$ can be repeated to arrive at a new solution using a different control

structure with an additional control input.

OEPI Method Results Re-solving the NLP problems that result from using $\boldsymbol{U}_a$ and $\boldsymbol{U}_b$ result in costs of 0.317 and 0.319, respectively. Both are better than the cost when using two parameters, but the control structure $\boldsymbol{U}_a$ results in a better cost than $\boldsymbol{U}_b$, as the OEPI method predicts. In practice, only a single control structure is used for each iteration of the OEPI method. However, for this study, multiple control structures are used and the results compared to experimentally verify the method.

The flowchart in Figure 24 shows a feedback loop between Step 6 and Step 3 for inserting variables one by one. The example in this section illustrated in detail each step of the process for a single iteration with the addition of comparing the other control structure options. This same process, including the comparison, was repeated for this problem as the number of control parameters increased and the results are shown in Figure 27(a). The cost is plotted against the number of parameters for different control structures. The control structures considered were generated by adding a single element to the previous best control structure. For example, there are two ways to generate a control structure of three element from the two-element control structure in Equation 42 (given the constraint of equivalence). The two resulting control structures are given in Equation 46 and labeled in Figure 27(a). The next three cost points, under "4" parameters, represent the possible control structures resulting from the addition of a single element to $\boldsymbol{U}_a$. The costs plotted using a box correspond to the control structure predicted by the OEPI method to yield the best result. The OEPI method accurately predicts which control structure will result in the best performance in every case. The prediction is carried out before any NLP problem using the additional degree of freedom is solved.

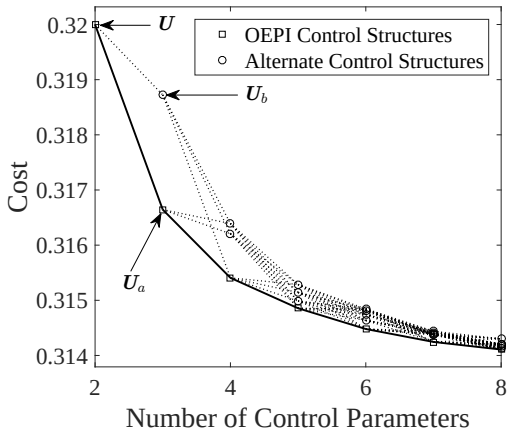
In Figure 27(a) the initial control structure had two equally spaced parameters, as seen in Equation 42. A control structure of three equally spaced parameters can never

result from the control structure in Equation 42 given the assumptions used in this method used to insert elements into the control structure. For this reason, the process is repeated using initial control structures of three and five equally space parameters and the corresponding results are plotted in Figures 27(b) and 27(c) respectively. Again, the control structures predicted to yield the best results are plotted using a box. An initial control structure of four parameters is not used as it would simply be a repetition of the control structures in Figure 27(a) from "4" parameters on. The OEPI method accurately predicts which control structure will result in the best performance for every case.

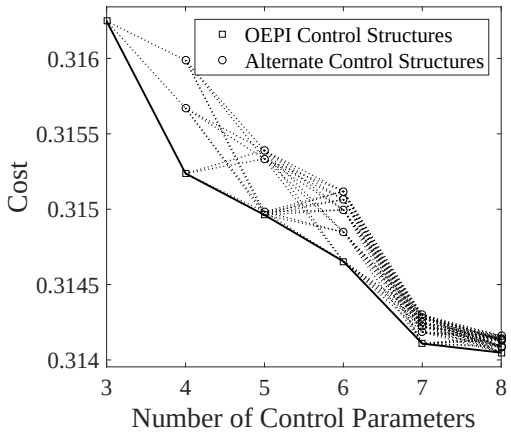
The benefits of using the OEPI method instead of arbitrarily selecting a control structure are clear from a cost improvement perspective, specifically in the context of trajectory optimization in conceptual design. Trade studies may require several trajectory optimization problems. When the direct shooting method is employed, the OEPI method is capable of generating control structures to minimize the cost. This is achieved in a traceable and repeatable manner without the need for an expert experienced in the problem at hand. This can save time and computational expense as well as increase confidence in the final solution. These benefits will be more clearly demonstrated in the following examples.

2.6.1.2 Optimal Launch to Lunar Orbit

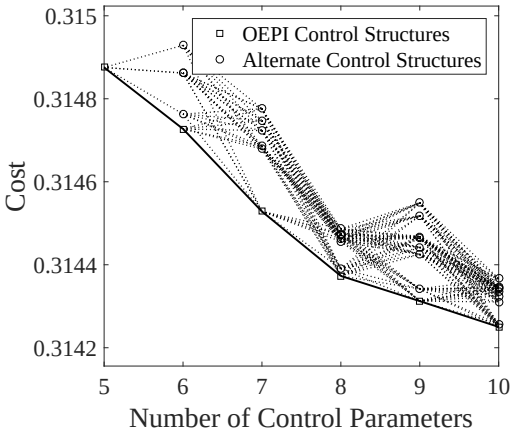
The time optimal launch of a satellite to orbit is a common problem in aerospace applications [104]. The second example problem for the OEPI method is the optimal launch from a lunar surface to orbit. This problem is more representative of the launch vehicle problems motivating this method than the previous problem. Now that the concepts have been demonstrated, they are tested on a more complex problem. The problem is set up as minimizing the time to orbit using constant acceleration under



(a) Two initial parameters



(b) Three initial parameters



(c) Five initial parameters

Figure 27: Cost vs number of control parameters with different initial control structures for the simple problem

a simplified set of dynamics (constant gravity and flat-moon approximations).

$$J = t_f \tag{47}$$

subject to the dynamics

$$\begin{array}{l} \dot{x_1} = x_3 \\ \dot{x_2} = x_4 \\ \dot{x_3} = \boldsymbol{f}\cos\theta \\ \dot{x_4} = \boldsymbol{f}\sin\theta - \boldsymbol{g} \end{array} \text{, where} \qquad \boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \tag{48}$$

Here, θ is measured from the horizontal and is used to control the vehicle. The state variables x_1 and x_2 are the range and altitude, respectively and x_3 and x_4 are the horizontal and vertical velocities, respectively. The parameters $\boldsymbol{f}$ and $\boldsymbol{g}$ are constants that represent the thrust and gravitational accelerations. The initial and final conditions are simply

$$t_0 = 0 \quad x_{10} = 0 \quad y_{20} = 0 \quad x_{30} = 0 \quad x_{40} = 0 \tag{49}$$

$$x_{2f} = h_{circ} \quad x_{3f} = V_{circ} \quad x_{4f} = 0 \tag{50}$$

where h_{circ} and V_{circ} are the altitude and velocity of circular lunar orbit.

The problem stated here has been solved analytically in Lungoski et al [104]. The optimal control guidance is

$$\tan\theta = at + b \tag{51}$$

where a and b depend on the specific problem and can be found numerically. This is the well known linear tangent steering law [125]. The solution method given by Longuski [104] is an iterative process using Equations 52 and 53. An initial guess is used for θ_0 and Equation 52 is solved numerically for θ_f. In Equation 53 the left hand side is dependent only on the specified problem, namely the vehicle variables and final constraints. The right hand side depends on the initial and final values of

the control θ. The values on both sides of the equality are compared, and the initial guess for θ_0 is modified until convergence. At that point the initial and final values of the control function θ are known.

$$\frac{\boldsymbol{f}}{\boldsymbol{g}} = \frac{\tan\theta_f - \tan\theta_0}{\sec\theta_f - \sec\theta_0} \tag{52}$$

$$\frac{2h_{circ}\boldsymbol{f}}{V_{circ}^2} = \frac{\tan\theta_0\sec\theta_f - \tan\theta_f\sec\theta_0 + \ln(\frac{\sec\theta_f+\tan\theta_f}{\sec\theta_0+\tan\theta_0})}{\ln^2(\frac{\sec\theta_f+\tan\theta_f}{\sec\theta_0+\tan\theta_0})} \tag{53}$$

The variables a and b used to define the control function are given by

$$a = \frac{\boldsymbol{f}}{V_{circ}}\ln(\frac{\sec\theta_f + \tan\theta_f}{\sec\theta_0 + \tan\theta_0}) \tag{54}$$

$$b = \tan\theta_0 \tag{55}$$

And the final time is

$$t_f = \frac{\tan\theta_f - \tan\theta_0}{a} \tag{56}$$

For more details on the derivation the reader is referred to Chapter 7 in Lungoski's text [104].

Analytical Solution The problem is solved specifically for the following vehicle variable values:

$$\boldsymbol{g} = 5.32\ ft/s^2 \quad \boldsymbol{f} = 15.96\ ft/s^2 \tag{57}$$

and the final conditions defined by a 100 nmi circular orbit:

$$h_{circ} = 607612\ ft \quad V_{circ} = 5511.3\ ft/s \tag{58}$$

This problem was solved iteratively using the equations in the previous paragraphs. The minimum time is 486.4 sec and the control equation is given in Equation 59.

$$\tan\theta \approx -7.7686 \times 10^{-3}t + 2.6116 \tag{59}$$

Numerical Solution using Direct Shooting This problem can be set up and solved using direct transcription. As in the previous problem, the control function will be approximated using a set of parameters. The dynamics and boundary conditions from Equations 48 - 50 remain unchanged. The optimization function $F(\boldsymbol{X})$, given in Equation 60, is set to the final time t_f, similar to the cost in Equation 47. The design vector $\boldsymbol{X}$ consists of three elements that when coupled with the control structure represent the control function u(t), as seen in Equation 61. Unlike the simple problem in Section 2.6.1.1, this problem does not have a specified final time. The final element in the control, u_3, applies until the trajectory termination criteria, in this case specified by the final velocity. The OEC in Equation 62 is introduced to enforce the constraints for final altitude and vertical velocity. The final condition for horizontal velocity is enforced automatically by using it as the trajectory termination criteria.

$$\text{Minimize} : F(\boldsymbol{X}) = t_f \tag{60}$$

$$\boldsymbol{X} = \vec{u} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}, \qquad \boldsymbol{U} = \begin{cases} u_1, & 0 \leq t < 175 \\ u_2, & 175 \leq t < 350 \\ u_3, & 350 \leq t < \textit{trajectory termination} \end{cases} \tag{61}$$

$$J = \text{OEC} = \left[(x_{2f} - h_{circ})^2 + x_{4f}^2\right] + t_f \tag{62}$$

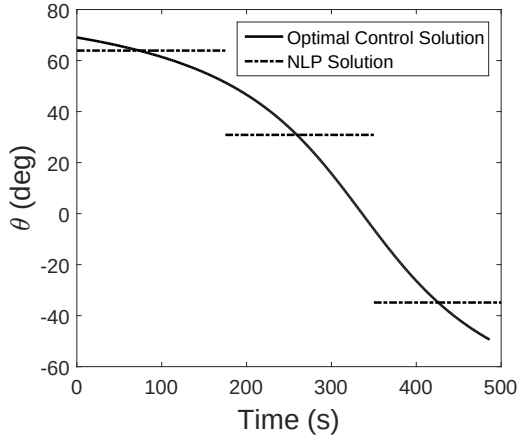
The standard unconstrained optimization function *fminunc* in MATLAB was used to solve for the parameter values [109]. A solution for this problem is given by $u_1 = 1.108$, $u_2 = 0.546$, and $u_3 = -0.614$. The resulting time to orbit is 493.8 s. The parametric control and state histories are shown in Figure 28 along with the optimal control and corresponding state histories. The difference in solution method is illustrated most obviously in Figure 28(a), but the effects are clearly seen in the velocity profiles (Figures 28(b) and 28(c)). This is because the control angle θ directly affects the direction of acceleration, which will result in discrete changes in the slope of the vertical and horizontal velocities. The altitude and range profiles, however, are

continuous and very closely match the optimal trajectories (Figure 28(d) and 28(e)).

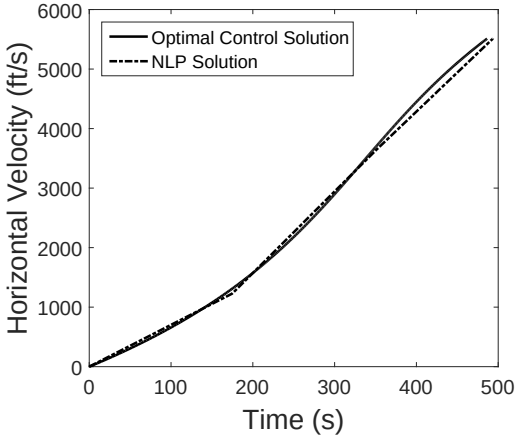
This parametric solution illustrates the power of the direct shooting method in the context of conceptual design. The total time to orbit for the parametric solutions was 493.8 s and for the optimal solution was 486.4 s. Consider the scenario where the optimal control solution was not known or not solvable. Direct shooting found a solution within about 1.5% of the global optimum by only optimizing three parameters. This solution was found with no knowledge about the optimal control function. In the following section, a solution within 0.26% of the optimal solution will be found by employing the OEPI method.

Application of the OEPI method and results Several iterations of the OEPI method were applied to this problem. The steps in Figure 24 will not be explained in detail here as they are described in detail in Section 2.6.1.1. The results from adding variables for the lunar ascent problem are shown in Figure 29. The indicated solutions, marked by a box, correspond to the control structures predicted by the OEPI method to yield the most benefit to the cost function. In every case but one, the OEPI method accurately predicts which control structure will result in the best performance.

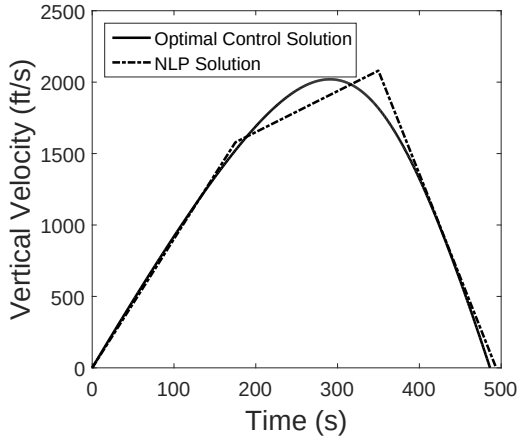
For four and five control inputs the method accurately predicts which control structure allows for the best solution. When six parameters are used, the best solution is found using a control structure that was not predicted by the OEPI method. The control structure predicted by the method yields a time to orbit of 487.85 s. Another control structure not predicted by the method yields a solution of 487.81 s. Even though the difference is very small, it does highlight a limitation of the OEPI method. The OEPI method is a numerical method based on information from the local region that the current best solution resides in. Moving away from that point may change how the cost function benefits from the replacement of a given parameter.



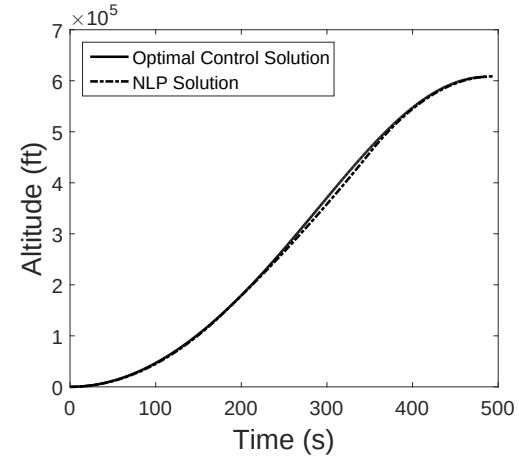
(a) Control History



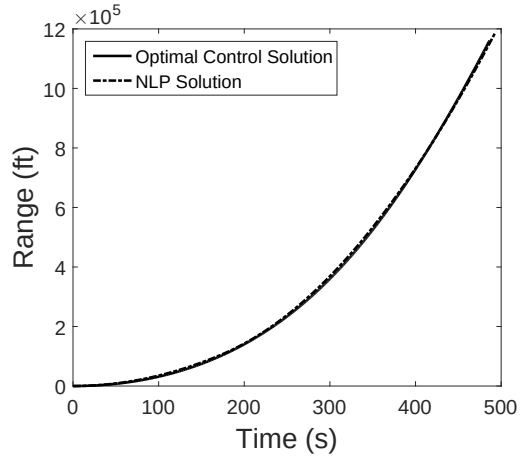
(b) Horizontal Velocity



(c) Vertical Velocity



(d) Altitude



(e) Range

Figure 28: Comparison of optimal and transcribed solutions for lunar launch problem

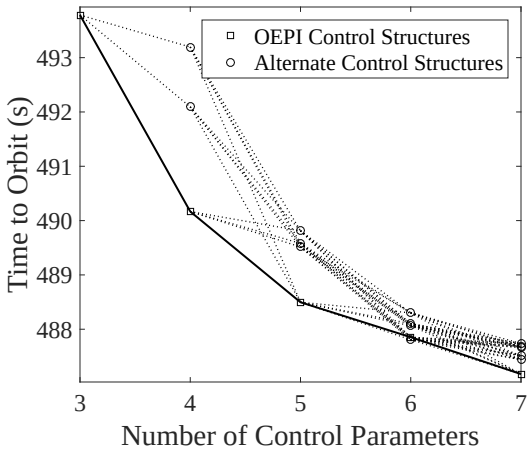


Figure 29: Cost vs number of control parameters for different control structures for lunar ascent problem

In this case the method is robust enough to find the correct control structure in the next iteration. When six parameters are used the control structure predicted by the OEPI method comes in second, and another control structure comes in first. However, both of these control structures predict the same subsequent control structure. If the OEPI method were followed, the optimal control structure using seven parameters would be found. In addition, the difference between the costs for these two six-parameter control structures in question is very small, less than 0.01%. The only way to determine this is to test all the six-parameter control structures available. This is time consuming and costly. The OEPI method provides a way of predicting which control structure will result in the most performance improvement without needing to test all the options. The OEPI method has been accurate so far in every case except one, and in the single case it was not accurate the error was one hundredth of one percent.

2.6.1.3 Launch Vehicle Trajectory Problem

The final example problem used to test this method is the optimal launch into LEO of the Delta IV Heavy, described in Section 2.5. The direct shooting method has been

used to solve problems in launch vehicle conceptual design [17, 19, 176]. The optimal solution is not known for this problem, so only the direct solutions are considered. In many conceptual design trade studies, optimized trajectories are required for multiple concept vehicles. Instead of relying on expert experience, the OEPI method provides an automated, traceable and repeatable method for generating control structures for these concept vehicles. This enables accurate performance measures to aid decision makers as the launch vehicle is designed.

The trajectory for the Delta IV Heavy shown in Section 2.5 is based on a control structure chosen *a priori*. This means it may not be optimal. To implement the OEPI method, the trajectory is solved using fewer parameters, and a control structure is developed by optimally inserting equivalent parameters. The program POST is used for the trajectory optimization.

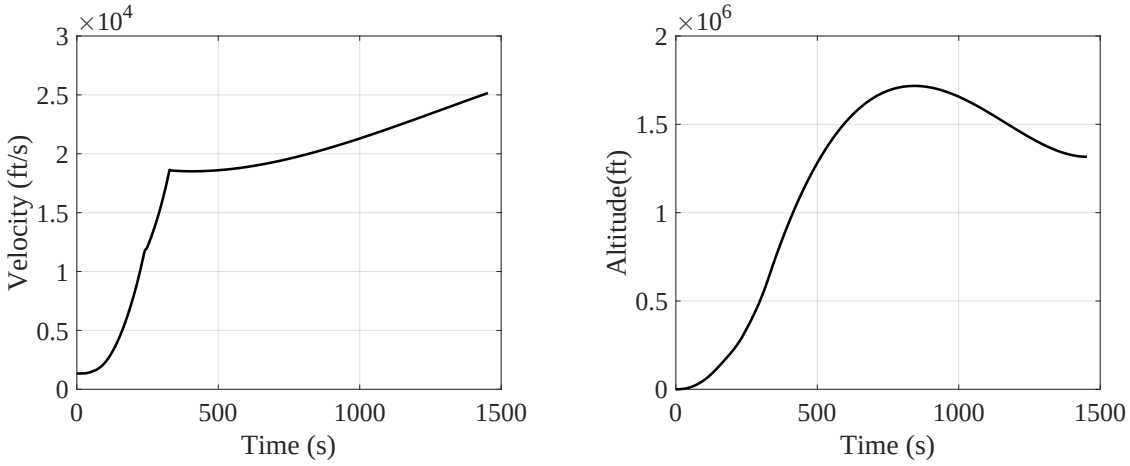
Numerical Solution using Direct Shooting Initially, four pitch rate values control the trajectory. These four parameters are approximating the optimal control function. Control input values, instead of rates, were used in the previous examples. The method remains unchanged, however, as the optimizer is acting in the same way regardless of whether values or rates are used. A fifth design variable is used for the launch azimuth. The design vector $\boldsymbol{X}$ contains five elements in this case. The first four, X_1 - X_4, are control inputs, u_1-u_4. The fifth element X_5 is simply a design variable, and not a control input. The OEPI method is relevant only to those design variables which are also control inputs, so it will not be applied on X_5. The four control inputs correspond to each of the major stages of flight: the gravity turn, powered ascent with boosters and core, powered ascent with the core alone, and powered ascent with the upper stage. POST internally creates an OEC to consider the objective function and relevant constraints (final altitude, final flight path angle, final inclination, and maximum dynamic pressure throughout the trajectory).

Figure 30 shows various profiles representing the solved trajectory. The vehicle reached orbit with a mass of 71375.6 lb, meaning there was 175.6 lb of available propellant after insertion. Although not necessary for this study, this evidences that the vehicle and mission are accurately modeling a Delta IV Heavy launch. Figure 30(a) shows the velocity vs time for the trajectory. Both staging events are seen as non-differentiable points in the profile, with the core staging event being much more prominent. The altitude profile in Figure 30(b) shows that the vehicle reaches a higher altitude before descending and inserting in the final orbit. This lofted trajectory was seen earlier in Section 2.5.1. The vehicle trades potential energy for kinetic energy as it descends and accelerates into the target orbit. This effect is seen in the flight path angle history in Figure 30(c).

Figure 30(d) shows the pitch rates used to control the vehicle. The thin lines at 0° pitch rate represent times in the trajectory where control inputs in the design vector $\boldsymbol{X}$ were not used to control the vehicle. These include the initial vertical rise to clear the launch tower and the zero angle-of-attack phase through maximum dynamic pressure. The vehicle is still subject to control at these times, but this control is not part of the optimization process and the values are not included in the plot. The first control input initiates the gravity turn and is only applied for a short time. Two other control inputs apply until the core is staged and a single control input is used during the upper stage flight.

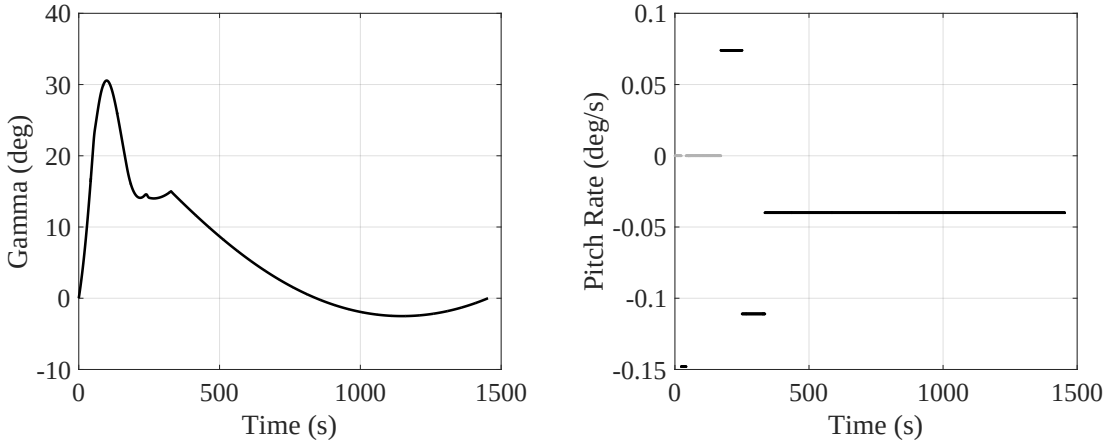
The trajectory discussed in this section was the best solution found using this control structure. Ten thousand repetitions were run with different initial guesses on the parameters to find the best trajectory. The following section will show how the OEPI method is applied and the control structure is updated for find better solutions.

Application of the OEPI method and results The OEPI method is applied to this problem (see Figure 24 for steps) starting with the control structure described



(a) Velocity Profile

(b) Altitude Profile



(c) Relative Flight Path Angle Profile

(d) Pitch Rate Profile

Figure 30: Delta IV Heavy launch trajectory profiles

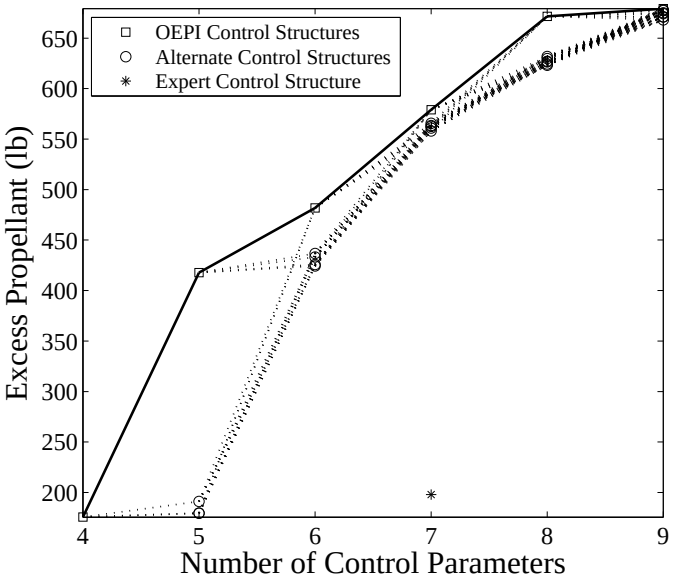


Figure 31: Cost vs number of control parameters for different control structures for Delta IV Heavy launch problem

in the previous section. Figure 31 shows the resulting costs for different control structures as the OEPI method is iteratively applied. In this case the goal is to maximize the metric (propellant remaining) instead of minimizing as in the previous problems. The control structures identified by the OEPI method outperform all the other control structures considered. In practice, if the OEPI method is applied, only six different control structures would need to be evaluated, one for each added parameter. For this study, the other control structures are evaluated for comparison purposes.

The benefit of this method is seen very clearly in the step between 4 and 5 parameters. The amount of propellant remaining more than doubles when the correct control structure is used. If one of the other two control structures are selected, however, only small benefits are seen. The selection of control structure will result in a 200 lb difference in the cost function by adding a single parameter. With the addition of each parameter, the OEPI method correctly predicts which control structure will yield the best performance. The control structures in the last iteration performed

for this study resulted in performances within about 10 *lb*. It would be inaccurate to conclude, however, that the control structures shown with nine parameters would result regardless of the control structure selections in previous iterations. The control structures plotted are all based on the previous best control structure. If the wrong control structures are selected in the iteration between four and five parameters, for example, the method would not arrive at the same control structures when nine parameters are used. In fact, each circle in Figure 31 could be used as the starting point for the OEPI method. For this study, only the best solutions were used as starting points. This illustrates the large number of options that could result when selecting a control structure. The OEPI method serves an important function as a method that accurately identifies the best control structures in a repeatable and traceable manner.

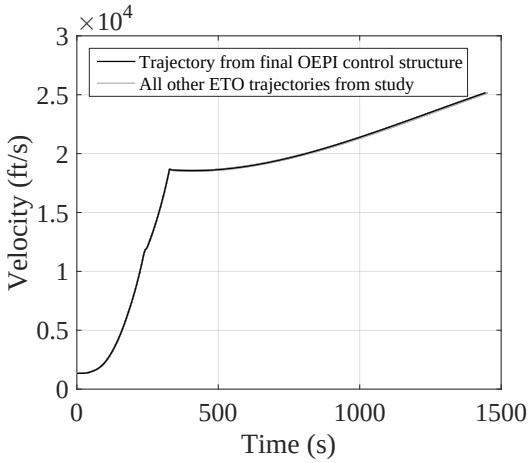
The control structure for four parameters is shown in Figure 30(d). The traditional method for determining control structures is for an analyst to select one *a priori*. Intuitively, it seems the problem would most benefit from splitting the last control variable into two, as it is applied for the longest amount of time (over 1000 seconds). This approach was carried out in a previous study using the same vehicle model for the Delta IV Heavy and the same target orbit. Seven parameters were used to approximate the optimal control function, and the best solution found resulted in 198 *lb* of propellant remaining [161]. This point is plotted with an asterisk (*) in Figure 31. The selection of control structure resulted in a 350 *lb* difference in the optimal answer. The OEPI method clearly selects better control structures than the analyst. The analyst intuitively chose to split the final parameter. However, when this control structure is considered it yields only small benefits. When the OEPI method is applied to this problem it accurately predicts that splitting the second parameter will yield the most benefit. The second parameter is split twice more, as the number of controls increases to 6 and 7, before it becomes more beneficial to split the final parameter.

Choosing the wrong control structure can lead to inaccurate approximations of vehicle performance. In conceptual design, trade studies are routinely used to compare vehicles based on performance [168, 176]. The wrong control structures result in inaccurate estimates of the vehicle's actual performance, as discussed in the previous paragraph. This can lead to selecting the wrong vehicles for further analysis, or eliminating the right vehicles. These mistakes will lead to cost overruns and schedule slips, and in an environment where costs are high and schedules are tight, this can result in canceled programs.

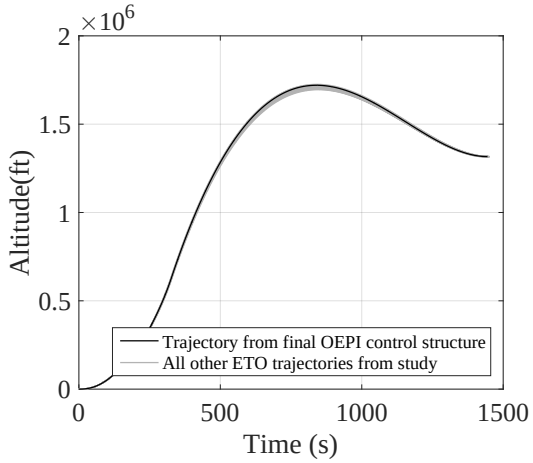
The resultant trajectories when different control structures are used are not significantly different. Figure 32 shows the trajectory corresponding to the best performance in a solid black line with all the other trajectories considered plotted in gray. There are some differences seen, especially in the altitude and flight path angle profiles in Figures 32(b) and 32(c). Each of the 31 trajectories in this plot correspond to a different control structure, and different performance. Traditionally, experts inspect the plots to determine if modifications to the control structure are required. A visual inspection is used instead of Steps $3-5$ in Figure 24. A visual inspection of the trajectory profiles, however, is not sufficient to determine if these trajectories would benefit from an additional parameter, and if so, where to add the parameter. The OEPI method is able to accurately identify where to add parameters to generate a control structure that most benefits the problem under consideration.

2.6.2 Answer to Research Question 2.1

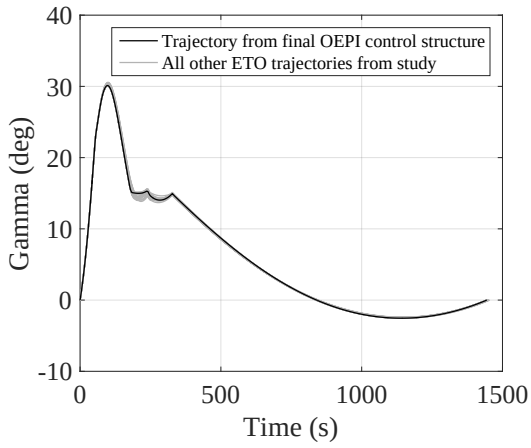
The OEPI method provides the answer to Research Question 2.1. It is a traceable and repeatable way to generate optimal control structures. In the world of launch vehicle conceptual design, this allows for consideration of new vehicles where subject matter experts may not exist and analysts don't have experience. In addition the OEPI method results in control structures that are superior to the control structures



(a) Velocity Profile



(b) Altitude Profile



(c) Relative Flight Path Angle Profile

Figure 32: Delta IV Heavy launch trajectory profiles for all control structures considered

used in other studies for known vehicles. Furthermore, the method is completely automatable. This enables continuous analysis even when analysts are otherwise occupied. As predicted by Hypothesis 2.1 in Section 2.6.1, a method based on the derivative difference can be applied iteratively to find control structures for these problems.

2.6.3 Application of OEPI Method to Design Space

The OEPI method provides a way of generating a control structure for a single vehicle. However, in this thesis, the goal is to consider the performance across a design space of vehicle design variables. This leads to Research Question 2.2.

Research Question 2.2 - How can this method of selecting control structures be applied to an entire design space?

The design space in question is described in Section 2.5.2. This continuous design space contains an infinite set of vehicles. Therefore, a representative set of vehicles will be selected from the design space to carry out the method on. The set will contain 21 vehicles; 16 will represent the extremes of the design space, 4 will be selected in the interior of the space using a LHC design and the final vehicle will be defined by the midpoint of the design variable ranges. The vehicle definitions for the 21 cases are given in Table 6.

The vehicles are selected as "reasonable" vehicles, meaning the variables describe vehicles that can fly. A vehicle with a thrust to weight ratio of less than 1 at lift off, for example, is not a reasonable vehicle. In addition, all the variables represent

Table 6: Vehicle definitions for the set of 21 representative vehicles

Vehicle	G_TW	US_TW	US_IMF	CB_ISP
1	1.16	0.17	0.05	400
2	1.35	0.17	0.05	400
3	1.16	0.50	0.05	400
4	1.35	0.50	0.05	400
5	1.16	0.17	0.12	400
6	1.35	0.17	0.12	400
7	1.16	0.50	0.12	400
8	1.35	0.50	0.12	400
9	1.16	0.17	0.05	450
10	1.35	0.17	0.05	450
11	1.16	0.50	0.05	450
12	1.35	0.50	0.05	450
13	1.16	0.17	0.12	450
14	1.35	0.17	0.12	450
15	1.16	0.50	0.12	450
16	1.35	0.50	0.12	450
17	1.18375	.45875	.05875	431.25
18	1.23125	.21125	.07625	406.25
19	1.27875	.29375	.09375	443.75
20	1.32625	.37625	.11125	418.75
21	1.255	0.335	0.085	425

continuous parameters. Based on this, Hypothesis 2.2 is given below.

Hypothesis 2.2 - If the design variables being considered are continuous and represent reasonable vehicles, then a usable control structure for the baseline will be usable for other vehicles in the design space.

A "usable" control structure in this thesis is defined as a control structure that will result in a change of less than 100 *lb* if a single control parameter is added using the assumptions in Section 2.6.1. The OEPI method provides a way of finding where to insert a parameter for the most benefit. Hypothesis 2.2 can be tested by using the baseline control structure, which was found in Section 2.6.1.3, and using it for the set of representative vehicles. The OEPI method can then be applied for a single iteration to determine how much benefit is seen from the addition of one parameter.

2.6.3.1 Baseline Trajectory Control Structure

The performance changes for the set of representative vehicles in Table 6 are not all less than 100 *lb* when a single parameter is added to the baseline control structure found in Section 2.6.1.3. This means Hypothesis 2.2 is rejected and another control structure is required. A method for finding a control structure that is usable for all the vehicles is developed and implemented in the next section.

2.6.3.2 Control Structure using Representative Vehicle Set

The control structure from the baseline vehicle was not a "usable" control structure for the set of representative vehicles. The OEPI method can be applied a different way, however. Previously, the OEPI method was only applied to single vehicle. Instead, all 21 vehicles can be considered simultaneously.

A result of considering multiple vehicles simultaneously is that each vehicle could benefit most from the splitting of different control parameters. This means multiple control parameters may be indicated as yielding the most benefit in a single iteration.

The goal of this experiment is to find a single control structure for all the vehicles in the design space in Table 5. It will be shown that all 21 vehicles in Table 6 can be modeled with a simple control structure using only four parameters. However, the addition of parameters to that control structure can change the propellant required to reach orbit of a vehicle by hundreds, and in some cases thousands, of pounds. This change is solely due to the control structure applied, not to any change in the vehicle itself. The control structure found in this experiment will be tested to ensure the change in propellant required due to the addition of a single variable is less than the required tolerance of 100 lb.

The method presented in the previous section is modified to apply to multiple vehicles. For the first iteration of the method (see Figure 24 for the iteration process) two parameters were added instead of a single one. This was done so that every vehicle could improve from the initial control structure. Since each vehicle called for a different parameter to be separated into two, the two parameters that were identified the most were chosen. It was shown, however, in the lunar example in Section 2.6.1.2 that even if a parameter is skipped in a single iteration, it will simply be called for in the next iteration.

After the first iteration, the parameter to be split was chosen based on the vehicles that improved the most in the previous iteration. The vehicles the improved the most carried were given preference in identifying the parameter to split. This process was used because in many cases, a single parameter was identified by many of the vehicles that did not improve by a significant amount when compared to the stated tolerance of 100 lb. Splitting these parameters would simply add complexity and yield little to no improvement to the problem.

At the end of the process, the control structure was verified for each vehicle by applying the OEPI process individually to each vehicle. Each vehicle's control structure was augmented by inserting the parameter selected by that vehicle's trajectory and the best performance was found. This ensured that the addition of a parameter to the control structure did significantly affect the optimization outcome.

In each of these iterations, the maximum performance was found by applying multiple initial guesses for the control parameters for each vehicle and optimizing the trajectory. In this study, 2500 control guesses were applied to each vehicle at each iteration.

Table 7 shows the performances for each of the vehicles with each control structure, labeled by how many parameters were in each control structure. Each control structure is the same for all the iterations except for the control structure with 13 variables, which is indicated with a * in the table. At this point, the optimal parameter split was done for each vehicle individually, therefore the control structure for each vehicle was different. It can be seen that the improvement from 12 to 13 for every vehicle is less than the limit of 100 lb. In fact the maximum difference is around half that and most vehicles improve by less than 10 lb.

Figures 33 and 34 plot the performance vs the number of parameters for the 21 representative vehicles defining the design space. At each of the parameters, the control structure is the same with the exception of the last iteration, which is indicated with a *. When 13 parameters are used, the control structure is determined individually by each vehicle using the OEPI method. The zero value on the vertical axis for the figures represents the vehicle's performance using the initial control structure (with four control parameters). This aids in visualizing the improvement seen for each vehicle. The initial performance is given in the caption of each figure. A visual inspection reveals the plots all look somewhat similar. Figures 33(a) and 33(e) show behavior similar to the example problem in Section 2.6.1.3. Figures 33(i) and 33(j)

Table 7: Application of OEPI method to set of representative vehicles

Vehicle	Number of Control Parameters								
	4	6	7	8	9	10	11	12	13*
1	-14188.431	-13527.031	-13195.863	-12997.313	-12891.494	-12879.477	-12834.178	-12834.519	-12824.277
2	8431.651	8632.367	8749.542	8791.864	8825.739	8825.658	8825.658	8847.274	8847.274
3	7405.241	7501.977	7536.833	7545.374	7564.604	7570.625	7570.626	7570.106	7574.243
4	12583.086	12657.659	12682.348	12694.95	12702.322	12710.105	12710.482	12710.482	12706.769
5	-31789.418	-29047.111	-28338.602	-27730.702	-27666.22	-27277.626	-27151.122	-27099.381	-27046.608
6	2706.931	2961.869	3071.135	3114.023	3174.089	3181.329	3194.496	3194.496	3222.808
7	4030.57	4132.92	4170.012	4183.706	4199.316	4209.169	4210.063	4210.063	4207.526
8	9542.986	9620.851	9649.191	9662.723	9672.55	9672.729	9671.142	9672.108	9672.944
9	24970.178	25051.289	25069.843	25225.822	25273.076	25284.782	25281.73	25281.73	25292.887
10	32993.196	33066.898	33119.952	33104.525	33146.761	33141.901	33145.984	33149.467	33142.727
11	19968.551	20079.007	20128.034	20152.494	20165.746	20168.861	20168.859	20174.176	20174.175
12	24169.036	24279.147	24302.521	24317.096	24322.52	24334.4	24333.665	24341.531	24337.005
13	20915.459	21077.929	21144.254	21262.599	21273.724	21280.96	21280.961	21293.204	21296.641
14	30395.77	30461.209	30546.513	30564.27	30576.061	30582.464	30582.465	30579.727	30579.786
15	17168.397	17283.271	17336.109	17361.144	17375.414	17377.197	17378.143	17382.047	17383.96
16	21716.635	21811.871	21846.893	21863.905	21873.074	21881.346	21884.063	21882.382	21883.594
17	16581.413	16682.403	16723.721	16752.516	16762.079	16765.386	16766.858	16766.858	16770.825
18	8542.8	8731.681	8783.48	8828.131	8841.358	8840.605	8849.169	8848.369	8855.595
19	24507.387	24611.949	24665.609	24677.404	24670.544	24681.41	24684.247	24691.164	24685.716
20	16468.527	16558.315	16598.372	16600.724	16619.596	16616.844	16616.844	16616.844	16616.45
21	18115.106	18226.019	18253.391	18292.444	18291.764	18293.67	18305.252	18297.59	18300.38

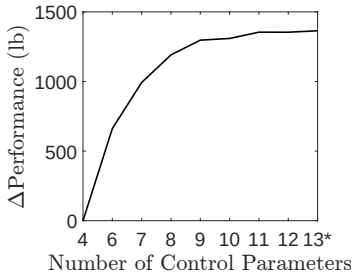
do show some different behavior in earlier iterations. This is because the control structure method was applied to 21 vehicles at the same time instead of individually to each vehicle. The vehicles that experienced the most improvement dominated the OEPI iteration process.

In the case of Vehicle 10, the dip in performance when 8 control parameters were used is a result of the global search. At each control structure, 2500 initial guesses for the control parameters are randomly selected. If a different 2500 guesses are selected, the results in Figure 33(j) may look different. This phenomena is seen for some of the other vehicles as well, albeit to a smaller degree. At any point on these plots, any performance seen with a lower number of control parameters is achievable by simply setting the values of the resulting split parameters equal to the value of the previous parameter. At worst, the performance should remain the same. In this case the search did not result in a better or even equal performance. The next iteration, however, shows a recovery of the performance.

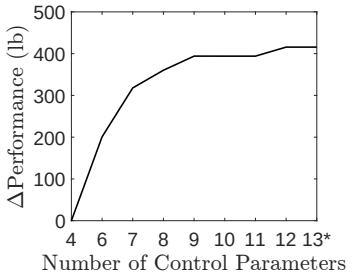
The plots in Figures 33 and 34 can be grouped by how much improvement is seen. Figure 35 shows the results for vehicles with large, average, and small improvements independently. The scale for ΔPerformance in each of these figures is drastically different.

A question may be raised regarding the methodology followed here for finding suitable control structures. In this work, a single control structure was found for the entire design space. This means the control structure has enough parameters to work well with the most difficult case, which were the vehicles represented in Figure 35(a). It also means that there may be vehicles that do not require as many parameters. Another option would be to find control structures individually for each vehicle. This would mean that each vehicle would potentially have a different control structure. This comparison is not considered in this study, but is left open for future work.

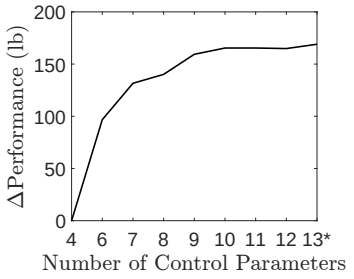
The hypothesis for this research question stated that if a usable control structure



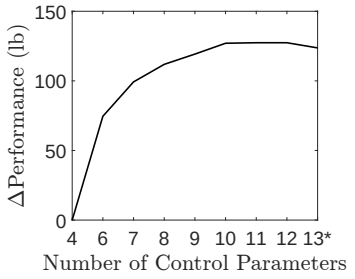
(a) Vehicle 1: -14188 *lb*



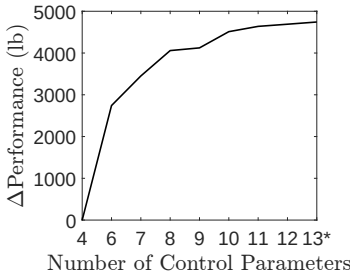
(b) Vehicle 2: 8432 *lb*



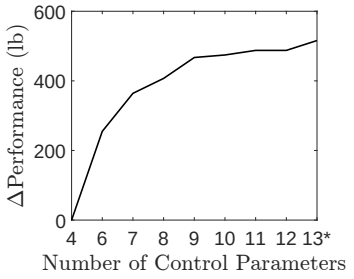
(c) Vehicle 3: 7405 *lb*



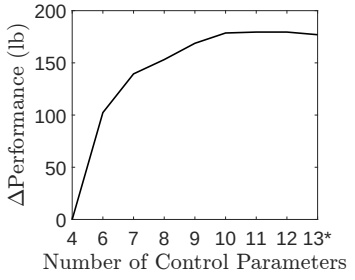
(d) Vehicle 4: 12583 *lb*



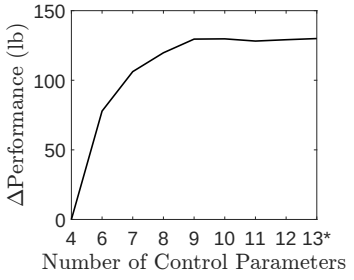
(e) Vehicle 5: -31789 *lb*



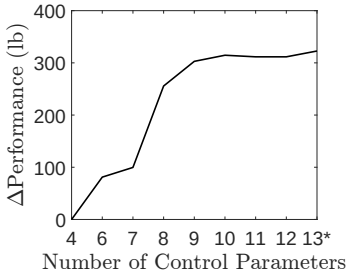
(f) Vehicle 6: 2707 *lb*



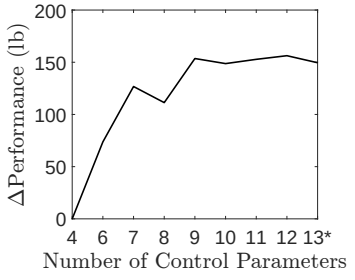
(g) Vehicle 7: 4031 *lb*



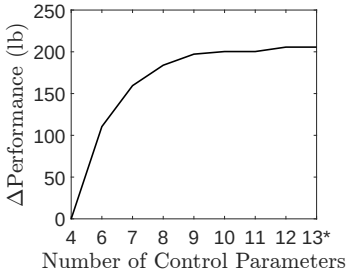
(h) Vehicle 8: 9543 *lb*



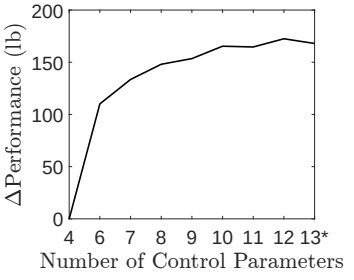
(i) Vehicle 9: 24970 *lb*



(j) Vehicle 10: 32993 *lb*

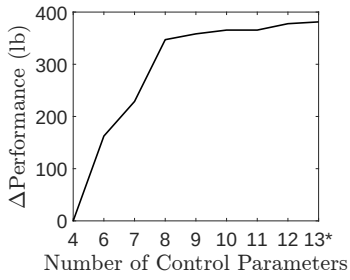


(k) Vehicle 11: 19969 *lb*

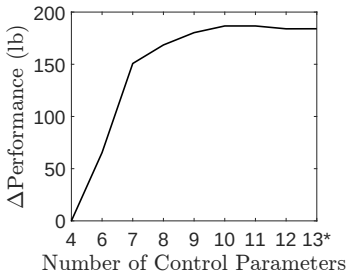


(l) Vehicle 12: 24169 *lb*

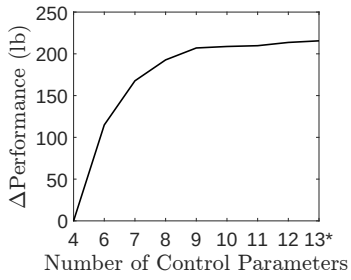
Figure 33: Change in performance (lb) vs number of controls for vehicles 1-12



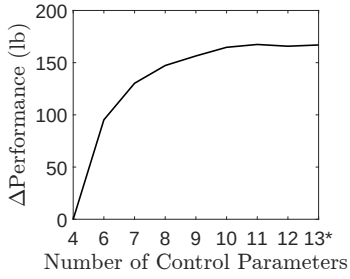
(a) Vehicle 13: 20915 *lb*



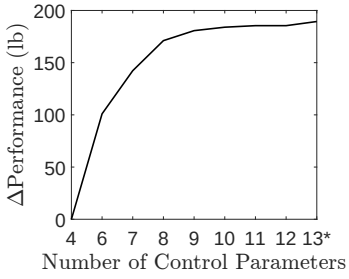
(b) Vehicle 14: 30396 *lb*



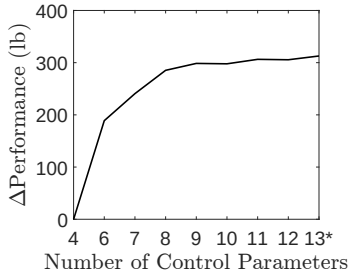
(c) Vehicle 15: 17168 *lb*



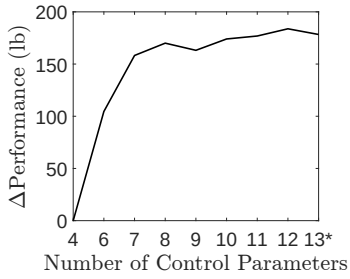
(d) Vehicle 16: 21717 *lb*



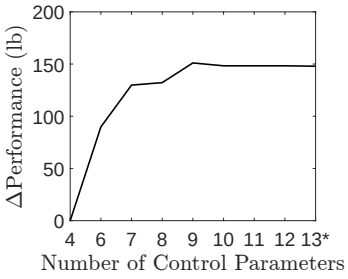
(e) Vehicle 17: 16581 *lb*



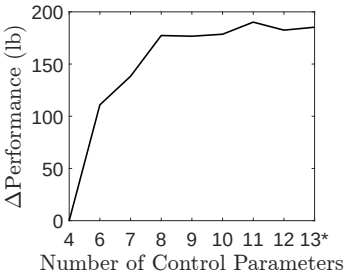
(f) Vehicle 18: 8543 *lb*



(g) Vehicle 19: 24507 *lb*

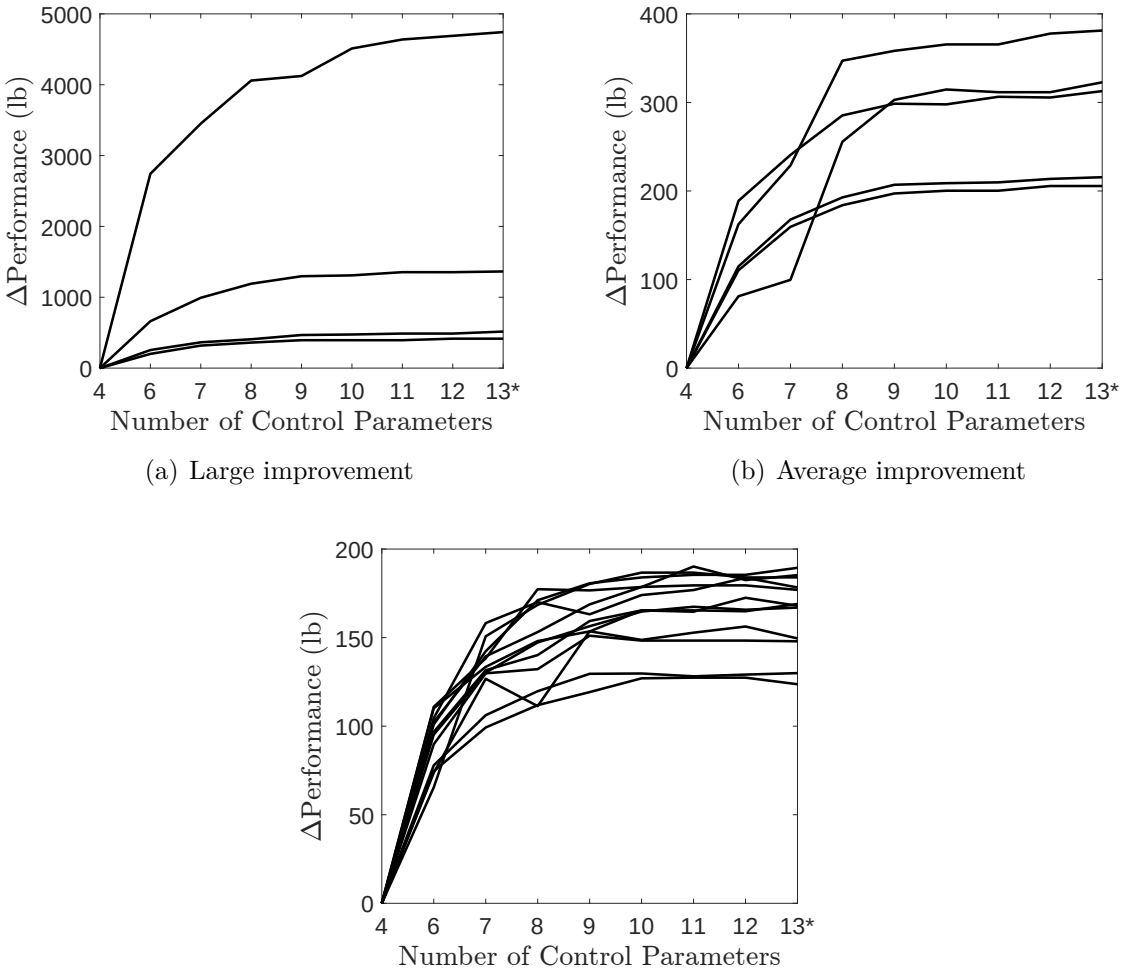


(h) Vehicle 20: 16469 *lb*



(i) Vehicle 21: 18115 *lb*

Figure 34: Change in performance (lb) vs number of controls for vehicles 13-21



(a) Large improvement

(b) Average improvement

(c) Small improvement

Figure 35: Cost vs number of controls for test vehicles grouped by magnitude of improvement

was found for a single vehicle, it could be used for other vehicles in the design space. This turned out to be false. However, a modified approach was to find a control structure that worked for all the vehicles at the same time. This resulted in a single control structure that applies to all the test vehicles. These test vehicles lie at the extreme corners, the exact center, and the interior of the design space. Because the control structure is successful for these vehicles, it is considered to be usable for the entire design space. For the rest of this study, this control structure will be used for any trajectory optimization unless otherwise stated.

2.6.4 Answer to Research Question 2.2

Hypothesis 2.2 proved to be false, as the control structure found for the baseline vehicle resulted in large performance differences with the addition of a single parameter for other vehicles. The answer to Research Question 2.2 is to apply the OEPI method simultaneously to a set of representative vehicles in the design space to find a control structure. Once this control structure is found, it can be applied to other vehicles within the design space to find performance values.

The control structure found in this experiment will be used for the rest of the analyses in this thesis. This control structure represents how the control parameters, in this case pitch rates, are applied to the trajectory. An example of two different control structures was shown in Figure 22. Optimization must still be performed to determine the specific values for each control parameter that result in an optimized objective function. This is addressed in Chapter 3.

2.7 Trajectory Optimization Conclusions

In this section the current trajectory optimization methods for launch vehicles were reviewed. Two research questions were developed and answered. The results are

summarized below.

Research Question 1 - Which optimization technique should be used to generate the performance data?

The first research question was answered from the literature. Indirect methods were eliminated because they could not be automated, an attribute necessary for the problem at hand. Direct methods combined with a global initialization were chosen because of their common use in trajectory optimization and applicability to this problem. Because the problem in question is a transcribed OCP, the optimization variables are parameters that represent a control function. The choice of how to represent the control function with a set of parameters led to the second research question, repeated below.

Research Question 2.1 - How should the control structure for a launch vehicle trajectory optimization problem be selected?

A method for selecting this control structure, termed the OEPI method, was developed and tested in Section 2.6.1. The OEPI method was developed specifically for the case when a small number of parameters is used to represent the control function, as is the case with launch vehicle trajectories. The method was demonstrated on three problems, including the optimal launch of the Delta IV Heavy. The OEPI method applies to a single vehicle, but the objective of this thesis is to consider a design space of vehicles. Consequently, a design space of vehicles was chosen for consideration in

this thesis. The design space is based on the Delta IV Heavy launch vehicle. The application of the OEPI method to this design space led to a subsequent research question, repeated below.

Research Question 2.2 - How can this method of selecting control structures be applied to an entire design space?

To answer Research Question 2.2, a set of launch vehicles were chosen to represent the design space in question. The OEPI method was applied to this set of representative launch vehicles, resulting in a single control structure that could be used for the design space. In the following chapter, a set of statistical methods for use in conjunction with the methods developed here is presented with the aim of enabling the research objective given in Section 1.3.

CHAPTER III

STATISTICAL METHODS FOR EVALUATING LAUNCH VEHICLE PERFORMANCE

Chapter 2 covered the relevant tools and methods for trajectory optimization. A combination of a global search with local direct optimization was chosen to generate launch vehicle performance data. The OEPI method was developed to address the challenge of representing the control function with a set of parameters. Figure 36 shows the elements of the method proposed in this thesis that have been covered up to this point. The result of this process in Figure 36 will be a set of performance values for each vehicle evaluated. However, only one performance value is required for each vehicle. Traditionally, the best performance is used, as seen in Figure 16. This approach can lead to inaccurate solutions when the data is used for surrogate models, as will be discussed in Section 3.2.1. Instead, methods from the field of statistics can be employed to leverage all the data available. This chapter will introduce several of these statistical methods for application in this thesis. To begin, a general introduction to the field of statistics is presented.

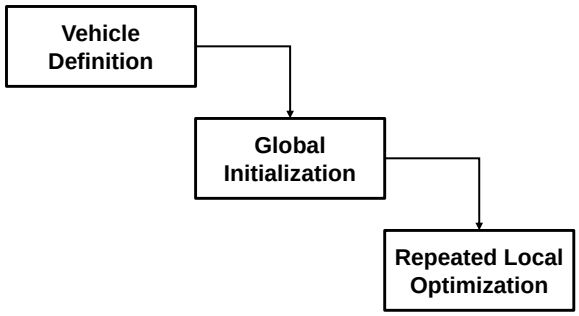


Figure 36: Method Buildup: Optimization method

3.1 Introduction to Statistics

Statistics is a field of mathematics that focuses on data analysis. While there is no set definition for statistics, areas of research included data collection, interpretation, presentation, as well as decision making based on the data [105]. Many times the terms statistics and probability are used together. Probability is a field of mathematics that tries to predict future outcomes. Statistics takes outcomes, or data, and attempts to glean as much information out of the data as possible.

Lurie gives a simple example of the difference between probability and statistics in his book *Applying Statistics* [105]. Suppose a jar contains a known set of 3 different colored marbles. If a single marble is removed, the likelihood of it being certain color can be calculated. If 10 marbles are removed, the odds of them all being the same color, or 5 of one color and 5 of another, can be determined. This is probability [105]. Suppose the same jar exists with 3 types of marbles, but the exact number of each type is unknown. If 10 marbles are removed, and 3 are one color, 2 are another, and 5 are the other, questions about the number of each of the 3 types of marbles in the jar can be answered. This is statistics [105].

In statistics there is a common set of terminology that will be briefly reviewed here. A population is a set of items to be studied. In the above example, the population is the set of all marbles. The individual elements of the population, in this example the marbles themselves, are items. A sample is a subset of the population. In statistics, the sample is used to find out information about the population. Both a sample and a population are a set of data. The difference lies in that a population is the set of complete data, while a sample is a subset of that set of complete data. In the marble example, it is possible to use the entire population, as there are a finite number of marbles. In many applications, however, it is impossible to get the entire population, as it is either infinite or too large. In these cases a sample is used.

There are several numerical measures that can be used to describe a set of data.

Examples of these numerical measures include the mean, median, variance, and many more. If the numerical measure is calculated using a sample, as opposed to a population, it is called an estimator. A few examples of these numerical measures are given below.

The mean describes the center of a data set. The calculation for the mean of a set of n units is given in Equation 63.

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{63}$$

The median is another measure of the center of a data set. It is defined as the "middle" of the ordered set. If there are an even number of items in the set, it is the average of the middle two.

The variance is a measure of how spread out the units of the set are. The variance is given in

$$s^2 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1} \tag{64}$$

Another useful measure for a data set is a percentile. The median is the 50% percentile. Likewise, the 25% or 75% can be calculated. In general, the p-th percentile is the value where at least $100 \cdot p\%$ of the items in the data set are at or below the percentile and at least $100 \cdot (1-p)\%$ of the items are at or above the percentile [86]. Percentiles will be used later on in this thesis to estimate the maximum performance value for a given vehicle.

3.1.1 Statistics in Trajectory Optimization

This section will discuss how statistics are useful for trajectory optimization. In Section 2.4 it was determined that direct optimization was the method of choice for this problem. But direct methods can get stuck in local optima and do not always find the global optimum. Consequently another method is required in conjunction

with direct methods for this problem. A global method is capable of overcoming this hurdle, as global methods are designed to find the global optimum. But the literature, reviewed in Section 2.3.3, showed that global methods are computationally intensive and therefore not an option for evaluating a large set of vehicles. The conclusion was to use a very simple global search to initialize a local direct method optimization for the trajectories. The global method is not used to carry out the optimization, but it is used to initialize the search.

What results from this approach is a set of performance values for each vehicle evaluated. This set of performance values is due to the optimization process, where values for the control parameters are being selected. This is not due to the control structure, discussed in Section 2.6. Figure 16 shows the process terminating in a single best performance. Immediately before that, however, there is a performance value for every single local search (assuming no code failures). This can be viewed as a distribution of performance values for the vehicle. This distribution does not represent the random behavior of the vehicle itself. It is a result of the optimization method. One may question why this method is chosen in the first place. The reason, explained in depth in Chapter 2, is because none of the other methods were found suitable to the application required here, namely an automatable and robust method capable of generating large amounts of performance data in the context and time frame of a conceptual design study.

One way to view this distribution of vehicle performances is to consider the trajectory a vehicle takes to orbit. In principle, there are an infinite number of trajectories a launch vehicle can fly to orbit. Figure 37 shows nearly $1,000$ different paths to orbit for a single launch vehicle. They all start and end at the same point, but take different paths. Some of the trajectories follow lines of constant energy height, given by $E_h = h + \frac{v^2}{2g}$ where h is height, v is velocity, and g is gravitational acceleration. Only

one of the paths shown, however, is the optimal[i] trajectory, as measured by propellant remaining, corresponding to the optimal performance. This trajectory is plotted in black in Figure 37. The set of trajectories, representing all possible paths to orbit for the launch vehicle, is referred to as the trajectory population with a corresponding performance population. The goal of any launch vehicle trajectory optimization method is to find the optimal trajectory and corresponding optimal performance. In conceptual design, it is desired to find this optimal performance quickly and for a large number of vehicles. The performance can then serve as a basis for comparison between vehicles. The global initialization with local direct searches employed in this thesis attempts to find the optimal trajectory but actually results in a subset, or sample, of the trajectory population with performances concentrated towards the extreme of the performance population. Statistics provides the tools to be able to infer information about the population with the sample [143].

An example of this sample of trajectories is shown in Figure 38 and the corresponding performance sample is shown in Figure 39. The trajectories are much more similar to each other than in Figure 37, but the performance sample in Figure 39 shows that the performance values still differ. Because the method for trajectory optimization used in this thesis, and commonly used industry, does not guarantee the global optimum, running a single optimization case is equivalent to randomly selecting a single trajectory from the subset shown in Figure 38. These trajectories all represent solutions optimized using the direct method. However, they are obviously not all the same. If a single optimization analysis is run, there is no way of knowing where the performance of the resulting trajectory would lie in the distribution of performances. It could be the maximum or it could be the minimum. The performance resulting from a single optimization run is equivalent to a random variable with a

[i]This trajectory is simply the best of the set, not the global optimum, but will be referred to as such for the purposes of this discussion.

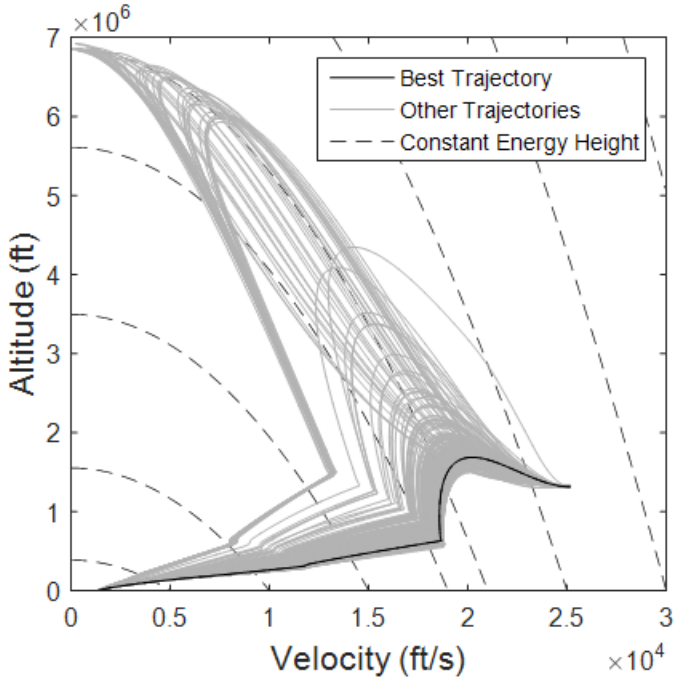


Figure 37: Trajectory population for sample vehicle

distribution equivalent to a normalized version of the histogram shown in Figure 39.

This presents obvious challenges in terms of finding a performance value for each vehicle in conceptual design. Consider the hypothetical situation where two vehicles (A and B) are compared. The global initialization with local direct searches was performed for both vehicles, resulting in two sets of performance values, like the one shown in Figure 39. Even though the method does not guarantee it, the analysis results in the globally optimal trajectory for vehicle A. The user, however, does not know this. Some good trajectories are found for vehicle B, but none are globally optimal. The reality might be that Vehicle B can perform better than Vehicle A, even though the maximum performance of each data set says otherwise. A metric that uses the entire set of data available, like a distribution percentile or the mean, may tell a different story. It may be of benefit, for example, to statistically estimate the 99th percentile or the 95th percentile of the sample or representative distribution. This may provide a more consistent estimate of how well the specific vehicle performs in the

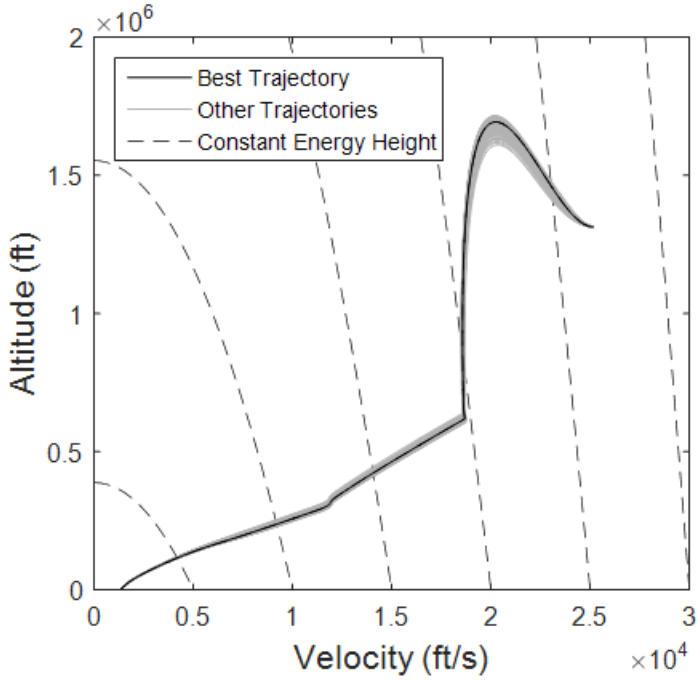


Figure 38: Set of optimized trajectories

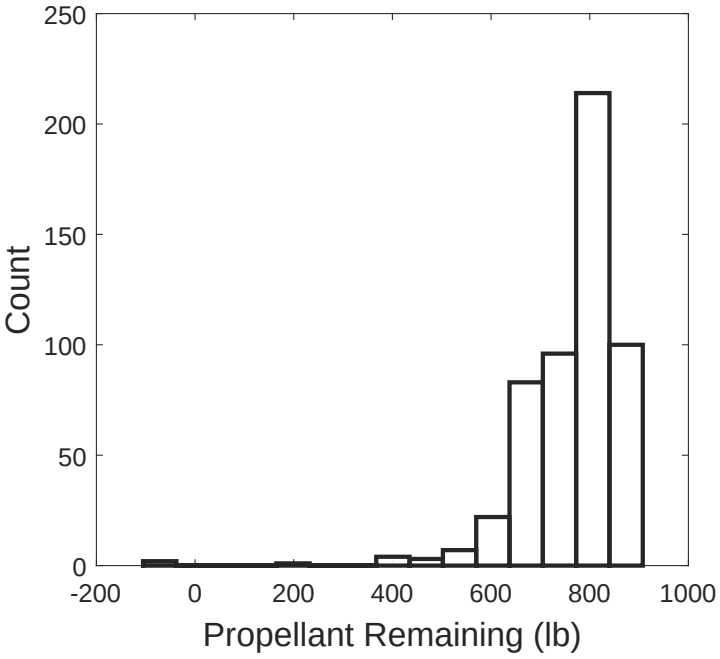


Figure 39: Performance histogram for the trajectories in Figure 38

context of the design space of vehicles being explored. The next section will present some methods of using the data obtained, namely the sample of items concentrated towards the extreme of the performance population, to obtain a single value for the vehicle performance.

3.2 Finding the Best Vehicle Performance

The aim of this section is to present some methods available for using the sample of performance values for each vehicle to find the maximum performance. A short aside is indulged here regarding the requirement for obtaining the maximum performance. Ideally, the trajectory evaluation method would yield the best performance in every case. However the tools necessary to do this in an automated fashion do not exist. Instead a combination of current trajectory tools and statistical methods will be linked to estimate the maximum performance.

The purpose of finding the performance values of different launch vehicles in conceptual design is to be able to make an informed decision regarding which vehicle designs should be carried forward to the next phase. Consequently, finding the globally optimal performance for one vehicle and a value 5% off the globally optimal value for the next vehicle can result in misleading conclusions. The consistency of the reported performance value is important in this context. It may be more beneficial to find the 95th percentile for all the vehicles, than to find the global optimum for some, and not others, while not knowing which are which.

An issue may be raised with this approach because the data represents something other than the absolute maximum performance of a vehicle. This must be viewed in the context of the method being developed. The ultimate goal is to generate a surrogate model for use in conceptual design. This surrogate model enables design space exploration of launch vehicles using the performance as a metric. Ideally, a surrogate model would give the numerical answer for each vehicle with 100% accuracy.

Unfortunately this is not feasible. Even if the surrogate model was fit using globally optimal data, the best surrogate models have some error. The conceptual designer must understand this and use the surrogate model accordingly. Surrogate models are very useful in design space exploration to identify particular vehicles that should then be explored in more detail using higher fidelity tools or analysis methods. Even if a surrogate model does not yield the exact numeric value of performance, it can still be extremely useful to find trends in the design space and identify regions of interest. A surrogate should never be used as a final answer to the vehicle performance. After exploring the design space and selecting one or a small number of vehicles, detailed and computationally intensive analysis tools will be used to get more accurate and confident estimates of performance. With this in mind, the following sections present several ways to estimate the performance for a given launch vehicle using a data sample. The first method presented is the sample maximum.

3.2.1 Sample Maximum

The simplest method would be to take the maximum of the sample. In this case, repetitions of the trajectory analysis are run for each vehicle and the best performance is reported. This method has advantages of being simple and easy to implement. In addition, there is no requirement on the number of repetitions required for this approach to work, meaning it can be applied with as few as a single case.

The situation where only a single case is used is analogous to randomly selecting a trajectory and corresponding performance from the set of trajectories shown in Figure 38. Without running more analyses, it is not possible to know where in that distribution the single item under consideration lies. The hypothetical situation described in Section 3.1.1 becomes a real possibility. Recall that this behavior is a consequence of the fact that the optimization result is dependent on the initial guess.

The disadvantages of this approach are mainly that there is little confidence in

that the global maximum is reached or in that a consistent valuation of the vehicle performances is achieved across vehicles. The confidence and consistency of the values will increase as more repetitions are run and the global search essentially covers the entire design space. As more cases are run, however, the time it takes to evaluate each vehicle increases. At a certain point the analysis will become computationally infeasible.

In the context of generating a surrogate model, it is very important to have consistent measures. Consider the situation where the first usable case is selected as the metric. If a surrogate model is generated using this data, it will result in trends that, in addition to being based on the underlying physics of the problem, will include trends based on random behavior resulting from the initial guess for the control parameters. The same outcome will result if the maximum is used, although it may be less problematic, especially as the number of analyses increases. The trends seen in the model will not only be based on the physics; in this case they would include behavior based on how close to the best performance is to the unidentified maximum.

3.2.2 Nonparametric Distribution Fitting

The last section presented options that used a single item from the sample to estimate the performance. Another option is to use every item available to generate a distribution and compute a metric from the distribution. There are two types of distributions that can be generated from a sample: parametric and nonparametric. A parametric distribution is one where the distribution is completely defined by a set of parameters [58]. For example, a normal distribution is defined by the mean and standard deviation. A nonparametric distribution uses statistical models that are infinite-dimensional [174]. In reality, an infinite number of dimensions are not used. A histogram, for example, is a nonparametric distribution that is defined using a series of bins. In theory an infinite number of bins could be used, but in practice

this is not the case.

A nonparametric distribution could be used to fit the performance data from the trajectory optimizer. Information about the underlying population distribution could then be inferred and extracted. Any distribution metric, such as the maximum or a percentile, could be used instead of the maximum of the data sample. Metrics from the distribution will be based on information from the entire distribution, rather than simply on a single data point.

There are several nonparametric distributions available for use. The simplest distribution, which has been referenced already in this thesis, is a histogram. A histogram splits the range of the metric of interest into a number m of bins B. For a normalized range $[0, 1]$ the first bin is $B_1 = [0, \frac{1}{m}]$. A histogram is then given by Equation 65.

$$\hat{f}_n(x) = \sum_{j=1}^{m} \frac{Y_j}{n \cdot h} I(x \in B_j) \tag{65}$$

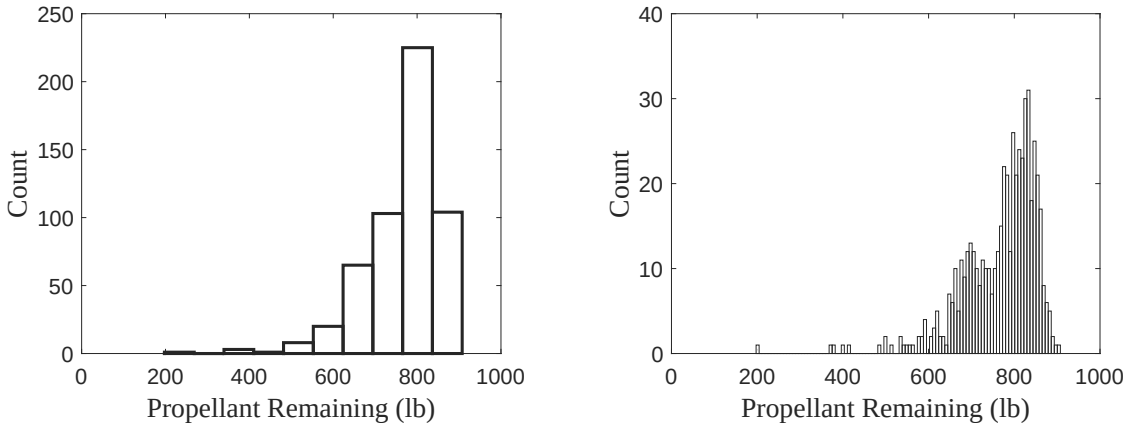
where h is the binwidth $1/m$, Y_j is the total number of observations in bin B_j and n is the total number of observations. The I function is given by

$$I(condition) = \begin{cases} 1 & \text{condition is true} \\ 0 & \text{condition is false} \end{cases} \tag{66}$$

This definition of the histogram is taken from Wasserman [174].

Appendix A contains an example set of sample performance data for a trajectory optimization problem. This data is plotted in a histogram in Figure 40(a). The histogram has $m = 10$ bins. The same data could be plotted using $m = 100$ bins, as seen in Figure 40(b). It is obvious that the bin size makes a difference in the final distribution.

For this problem there is no benefit to using a histogram over the sample maximum, described in Section 3.2.1, if the distribution maximum is used. The maximum of a histogram would be identically the maximum of the sample, and therefore there



(a) 10 bins (b) 100 bins

Figure 40: Example histograms of sample launch vehicle performance data

is no need to create the histogram. Histograms are useful, however, in understanding other nonparametric distributions.

Kernel density estimation (KDE) is another type of nonparametric distribution. KDE provides a smooth distribution to fit the given data. Unlike the histogram, it uses a smoothing function. In general, a kernel density estimate will converge to the actual distribution faster than a histogram [174]. A kernel estimator uses an input distribution along with the data to smooth out the fitted distribution. Like histograms there is a binwidth h that is applied to the data. Equation 67 describes a KDE.

$$\hat{f}(y) = \frac{1}{n} \sum_{i=1}^{n} w(y - y_i; h) \tag{67}$$

where n is the total number of samples and w is the input distribution [25].

The choice of input distribution is an assumption regarding the fitted distribution, but luckily, that choice turns out to be quite unimportant [174]. The choice of binwidth, however, does play an important role.

Figure 41 shows a histogram of the data in Appendix A along with two KDE distributions. It can be seen that the distribution with more bins, i.e. smaller binwidth,

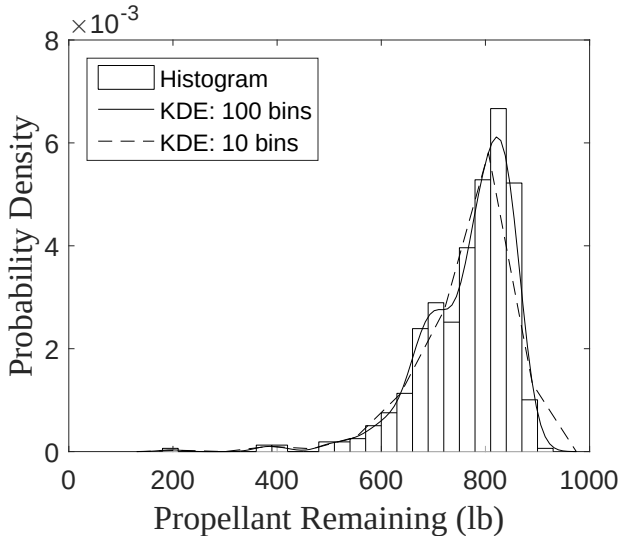


Figure 41: Example KDE distribution of sample launch vehicle performance data

is more accurate. It is also interesting to note that the maximum of the KDE does not necessarily equal the maximum of the histogram. Using KDE provides a different estimate for the population maximum than a histogram for the same sample data.

There are several other nonparametric distribution options. Variable binwidths are similar to KDE, but the binwidth is calculated for each point [25]. Orthogonal series and local polynomial methods can also be employed [174].

3.2.3 Parametric Distribution Fitting

In parametric distribution fitting the form of the population distribution is assumed to be known, and the parameters defining that distribution are sought after [143]. Several methods exist to estimate the parameters given the data to be fit. The method of moments matches statistical moments of the assumed population to the moments of the data sample by varying the parameters. Maximum likelihood methods find the parameters that lead to the highest probability of data sample being chosen from the population. Other methods include Bayesian methods, least squares method, and minimum distance method [143].

Any parametric distribution can be used. A distribution fitting software will have

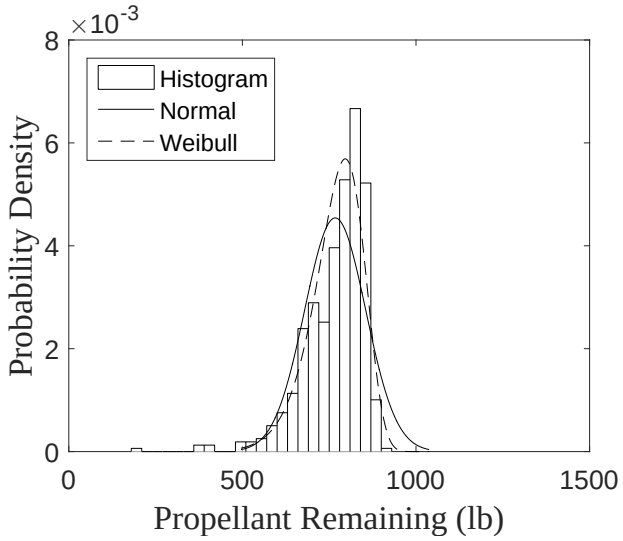


Figure 42: Example parametric distributions of sample launch vehicle performance data

an assortment of available distributions to consider when fitting the data, including beta, exponential, gamma, normal, Poisson, and uniform [84]. The challenge is selecting the correct distribution to assume for the population of interest. If nothing is known about the population it may be useful to consider all the distribution forms available and measure how well each form fits the sample data. The particulars of how goodness of fit tests are carried out are not discussed here. For further information the reader is referred to Olivares [113]. This approach is implemented in a MATLAB function and could readily be translated to an open source software [147]. Figure 42 shows the data from Appendix A along with 2 example parametric distributions fit to that data: the normal distribution and the Weibull distribution.

There is a danger when using parametric distributions to fit data. It implies that the population from which the data was sampled is of a certain form. Figure 43 shows the sample data with a Beta distribution fit. Obviously this selection of distribution is a bad choice for this data. But the software was still able to output a fit, so it is important to be able to confirm that assumed form is a good fit.

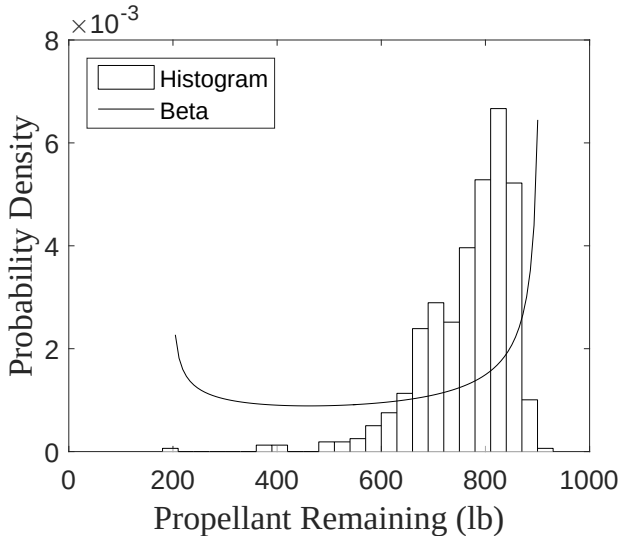


Figure 43: Example of bad parametric distribution fit of sample launch vehicle performance data

It is not uncommon that the distribution of a sample will look unlike the distribution of the population. As the sample size increases it will approximate the actual population more and more. If information about the population distribution is known it can inform the class of parametric distributions to be used to fit the data. The goal is not to find the distribution that best fits the sample data, but to find the distribution from which the sample data comes. The question is, what kind of parametric distribution, if any, does the data for the problem addressed in this thesis come from? The following section considers this question.

3.2.4 Extreme Value Theory

The method selected to obtain performance data for a given vehicle consists of using a global search to initialize several repetitions of a direct method optimization routines (see Section 2.4). What results is a distribution of performance data similar to the distribution shown in Figure 39. This distribution represents the performances of a sample of trajectories from the population of all possible trajectories, shown in Figure 37. It turns out the distribution of performances has some special characteristics that

can be exploited to result in a performance metric for each vehicle.

Generally statistics is interested in how the majority of a population will behave. In this thesis, the interest lies at the upper end of the performance distribution. The statistics introduced in Section 3.1 are not good measures of what is happening at the ends of the distribution. This is where Extreme Value Theory (EVT) is of great use.

> "The aim of a statistical theory of extreme values is to analyze observed extremes and to forecast further extremes." [71]

EVT started in 1928 with a paper by Fisher and Tippet investigating the frequency distributions of sample maxima/minima. In 1958 Gumbel published his book "Statistics of Extremes" [71] and it has for a long time been considered the standard reference for practical use of extreme value theory. Most of the discussion in this thesis is based on a more recently published book from 2004 by Beirlant, et al., called "Statistics of Extremes: Theory and Applications" [16].

The mathematical motivation for EVT is developed here. Assume a population distribution F. The maximum of a sample taken from F is given in Equation 68.

$$X_{n,n} = max\{X_1, X_2, ..., X_n\} \tag{68}$$

If several samples are taken, the distribution of $X_{n,n}$ can be considered. As $n \to \infty$ the distribution will become degenerate (i.e. the distribution will be a single value), because each sample maximum will be the maximum value of F, the underlying distribution. In practical problems, however, n does not to go ∞.

A normalization with parameters a_n and b_n can be introduced such that

$$P\left(\frac{X_{n,n} - b_n}{a_n}\right) \to G(x) \tag{69}$$

Equation 69 is taken from Beirlant [16], although it is also developed in other forms by de Haan and Ferreira [53] and Gumbel [71].

EVT is concerned with answering two questions. The first is finding all the possible the distribution functions $G(x)$. This is called the extremal limit problem, and leads to the extreme value distributions [16, 53]. The second is finding for which distributions F do the extreme value distributions apply. This is known as the domain of attraction problem [16].

Both of these problems have been solved, and the results are taken and applied in this thesis without the derivation. The reader is referred to Beirlant or de Haan and Ferreira for more details [16, 53].

It turns out the extreme value distributions are all of the form

$$G_\gamma(x) = exp\left(-(1+\gamma x)^{-1/\gamma}\right), \qquad \text{for} \quad 1+\gamma x > 0 \tag{70}$$

Here the value x is the normalized maxima of the samples. A slightly more general formulation is given in Equation 71 where x is not normalized.

$$G(x;\sigma,\gamma,\mu) = \begin{cases} exp\left(-(1+\gamma\frac{x-\mu}{\sigma})^{-1/\gamma}\right) & 1+\gamma\frac{x-\mu}{\sigma} > 0, \gamma \neq 0 \\ exp\left(-exp(\frac{x-\mu}{\sigma})\right) & x \in \mathbb{R}, \gamma = 0 \end{cases} \tag{71}$$

This is known as the Generalized Extreme Value (GEV) distribution [16]. The parameter γ is called the extreme value index (EVI), and σ and μ are the normalization parameters, equivalent to a_n and b_n from Equation 69 respectively [53]. Figure 44 shows the GEV distributions for different EVI values.

What EVT shows is that the distribution of sample maxima, where the samples are from a population distribution in the domain of attraction of the GEV distribution, will result in a GEV distribution. For example, the normal distribution is in the domain of attraction of the GEV distribution. If a sample is taken from a normal distribution, that sample will have a maximum. Five hundred samples will result in 500 sample maxima. The distribution of these sample maxima will be a GEV distribution. This process was implemented using the normal distribution shown in Figure 45(a) and the resulting GEV distribution is shown in Figure 45(b) with the

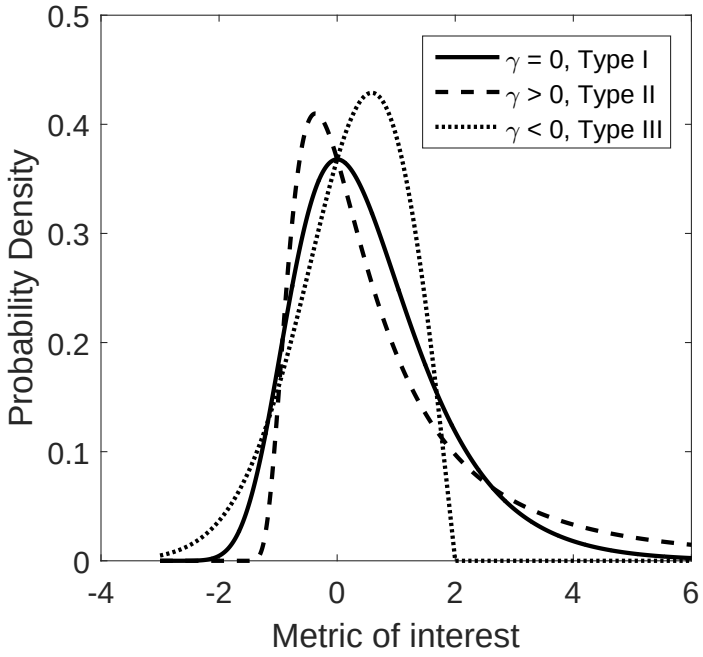


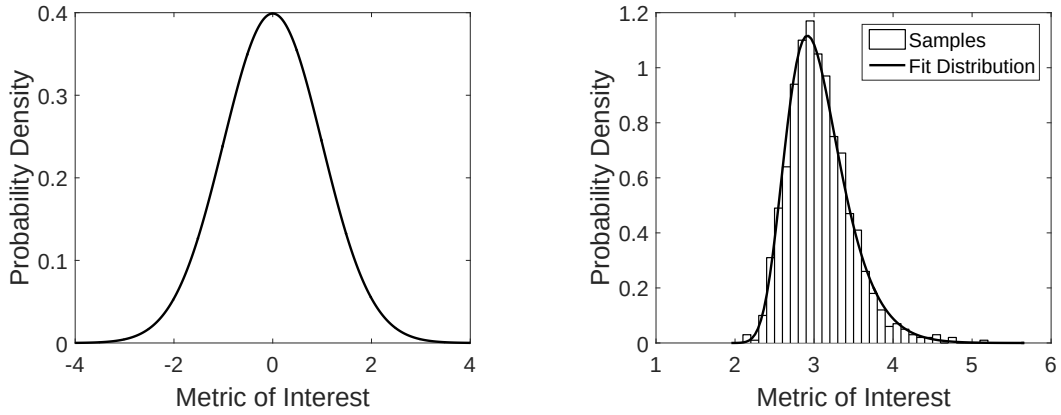
Figure 44: Example generalized extreme value distributions for different values of the EVI (denoted by γ)

corresponding data.

3.2.4.1 Extreme Value Theory Applications

EVT has been applied in several domains since its inception, including hydrology, meteorology, financial markets, and insurance. Examples in meteorology include predicting the probability of an extreme event, such as high precipitation. Smith [152] analyzes the record amount of snow precipitation that occurred in North Carolina on a single day in 2000. Events such as these cause massive disruptions in travel, transportation, power and communication lines, and school systems. Predicting the probability an event like this, and the uncertainty associated with the prediction, can lead to better preparation in future events.

Predicting extreme events in the context of weather examples can be carried out using historical data. In the case of the North Carolina example, the maximum amounts of snow fall from previous years can be used to determine the distribution of



(a) Population distribution

(b) Distribution of sample maxima

Figure 45: Normal distribution and corresponding distribution of sample maxima

sample maxima. In this case, the sample maxima is the largest amount of snow fall in a given day every year. This sample represents all the possible amounts of snow fall, and each year will result in a single sample maxima. The distribution of sample maxima created using multiple years can then be used to determine what the record snow fall will be for a given year. These record snow fall values will help develop contingency plans for these events.

Similar studies have been done with ozone extremes with respect to pollutants [152]. There have been efforts to reduce the amount of ozone present at ground level, which is harmful to humans (not related to the ozone in the upper layers of the atmosphere that helps block radiation). However, ozone levels are a function of regular weather patterns as well as the presence of certain pollutants produced by industrial activity. Using EVT, the maximum ozone levels due to regular meteorological phenomena can be quantified, and the effect of human activity on the ozone levels can be distinguished, and therefore regulated.

Studies considering rainfall and wind speed have been used to quantify effects of global warming and determine a hypothesized trend of increasingly extreme weather patterns [152]. Extreme rainfall data has also been used to study patterns in Florida,

South Korea and Taiwan [118]. These have applications in predictions for flooding and for establishing requirements for construction.

It should be noted that EVT does not provide a way of predicting when a maximum event will occur. For example, the model of North Carolina snow fall does not provide a way of predicting what day in 2020 will receive the most snow fall. The utility in these models is being able to predict the likelihood of receiving some record amount of snow fall in the next 100 years, for example. This is useful in several respect. Take road construction for example. North Carolina may decide to ensure their roads survive a 10 year snow storm (record snow fall predicted in a 10 year period), but not a 500 year snow storm.

Meteorological studies may be some of the first uses of EVT, but recently financial applications have become very common. Stock market predictions and insurance analysis are two of the most common applications. The usefulness of EVT for an insurance company can be exemplified by using the rainfall examples discussed previously. If an insurance company provides flood insurance, it is very useful to be able to predict what the extreme rainfall values will be, even if the particular day of extreme rainfall is not known. Another application for insurance companies is to examine the largest claims over a certain period of time. This can be used to set premiums as well as to determine if the largest claim values are changing over time. An example of this type of study is provided by Smith [152].

Another example of the application of EVT to financial industries is the prediction of stock market activity. The stock market represents great financial opportunity but can lead to large losses as well. A key measure known as Value at Risk (VaR) is used to quantify potential losses due to poor market behavior over a certain time frame [43]. EVT can be combined with this measure to determine the worst case scenario for a particular investment. As with the meteorology examples, EVT does not provide a way of predicting the market performance on a particular day. However, it is "useful

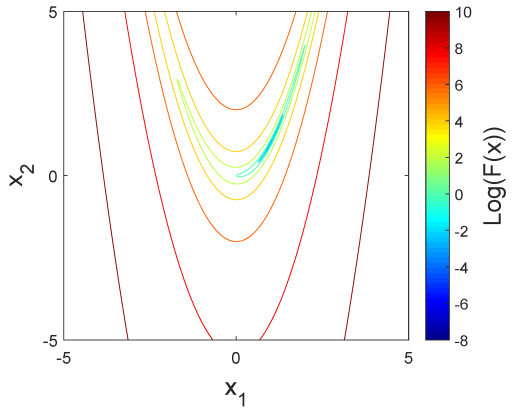


Figure 46: Contour plot of the Rosenbrock function using a log scale

for assessing the size of extreme events." [66]

3.2.4.2 Extreme Value Theory and Optimization

For this thesis, EVT will be applied in the context of optimization. An optimization process can be viewed as finding the extremal item in an underlying performance distribution. Consider the Rosenbrock function, given in Equation 72 and shown in Figure 46. The contour lines are plotted using a log scale. If a random point (x_1, x_2) is evaluated, there is some probability that it will result in a function evaluation between 10^2 and 10^4. This is true for any of the function values shown in the figure. The probability density function can be plotted, as seen in Figure 47. This represents the population distribution of function values for this problem. EVT is interested in the values at the extreme ends of this distribution, in this case the left tail (the goal is to minimize the Rosenbrock function). The goal of an optimization process is to find minimum value of the function, which is the item at the extreme of this population distribution. This leads to the third research question.

$$F(x) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2 \tag{72}$$

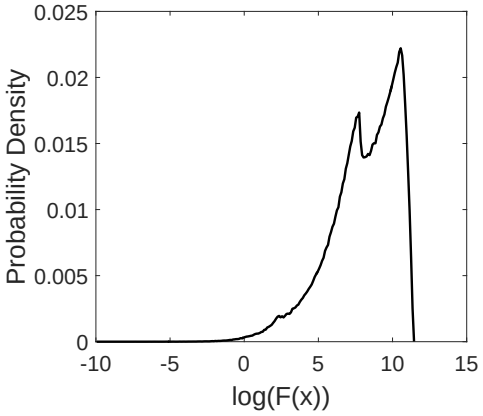


Figure 47: Performance distribution (log scale) when a random search is applied to the Rosenbrock function

---

Research Question 3 - Does EVT apply to optimization problems?

This question has been asked previously in the literature [12, 80], but not in this context. For the sake of this discussion, however, consider a random search on this function. This is applied by randomly selecting a set of points and evaluating them, resulting in a set of function values. This set of function values can be thought of as a sample from the population distribution (shown in Figure 47), and the minimum value of the random search as the sample minimum. If this is true, then repeated random searches, each with a minimum, should result in the GEV distribution. This

leads to the hypothesis for this question, stated below.

Hypothesis 3 - If a random global search is performed multiple times to find the optimum of a function, then the resulting distribution of best results will be a GEV distribution.

It should be noted that EVT literature usually deals with sample maxima, but the same theory applies to minima, just as optimization theory applies to maximization and minimization problems.

3.3 Application of Extreme Value Theory to Optimization

EVT has been introduced in the context of optimization in the previous section. EVT deals with the values at the tail end of a distribution. These values result from taking the maximum (or minimum) value of a sample multiple times to result in a distribution of sample maxima (or minima). Similarly, a random search can be considered as taking a sample from the distribution of performance indexes. The distribution of maxima (or minima) of repeated random searches should then result in a GEV distribution, as predicted by EVT.

In this section, Hypothesis 3 will be tested using an experiment. A similar study was published by Hüsler regarding the application of EVT to optimization [12, 80]. Hüsler's work focused on the applications of different stochastic optimizers, but he did conduct a random search on three test optimization functions. The random searches consisted of $n = 600$ points, and were repeated $k = 500$ times. The 600 points were generated using a normal distribution, and several different normal distributions were tested. A goodness of fit test was used to determine if the results were a GEV distribution. The results were not repeated, so only one p-value was available for

each test. These p-values indicated inconclusive results. However, Hüsler did provide an analytical approach to support the application of EVT to optimization. The remainder of Hüsler's work focused on other stochastic optimizers.

There are several differences between the work Hüsler published and the work being done here. In this case, the goal is to show the successful application of EVT to optimization. Therefore, the random searches will be generated using uniform distributions across the design space. The random searches in Hüsler's work assumed knowledge of the optimum value and the points were normally distributed near the optimum. A second difference is that the number of points in a random search can make a difference in the distribution of the sample optima. Hüsler used $n = 600$ points. Here, $n = 5000$ points will be used. This can be understood by considering the extreme option of using $n = 1$ point for a random search. The resulting distribution of sample optima, which will simply be the single point in each random search, will represent the distribution of the performance index values in the design space sampled. There is no reason to believe that this distribution will follow any pattern. As n increases, the GEV distribution will emerge if EVT is applicable to optimization. For that reason, a higher number of points in the random search is chosen. The final difference is that in this case several different test functions will be considered. Hüsler considered three simple functions. In this experiment, these functions along with several others will be tested. These will be described later in the section. The ultimate goal is to apply EVT to trajectory optimization when the direct method is used, so test functions will be chosen to represent some of the characteristics of that problem.

3.3.1 Experimental Setup for Research Question 3

The experiment to test Hypothesis 3 will be conducted by applying repeated random searches to a set of test function. The list of test functions will be presented in

Section 3.3.1.1. Each point in the random search will be evaluated, resulting in a set of performance values. The random search is being considered analogous to a sample, and therefore the minimum performance value of the random search will be analogous to the sample minimum. Repeated random searches will result in a distribution of sample minima. A GEV distribution can be fit to the distribution of sample minima and the goodness of fit can be evaluated. The minimum is used here because most optimization test functions are formulated as minimization problems.

For this experiment, two goodness of fit tests will be used. The first is the chi-square test. The chi-square test can be applied to any distribution so long as the cumulative distribution function (CDF) can be calculated [120]. A set of data and hypothesized distribution are compared by dividing the metric of interest into several bins that cover the entire set of values. The frequency of results in each bin is then compared to the expected frequency from the hypothesized distribution. This is done using the chi-square statistic, shown in Equation 73 [105].

$$\chi^2 = \sum_{bins}^{max} \frac{(Observed - Expected)^2}{Expected} \tag{73}$$

The value of this statistic is then compared to the distribution of the chi-square statistic when samples are taken from the hypothesized distribution. The p-value indicates what percentage of samples taken from the hypothesized distribution resulted in a chi-square statistic value greater than the sample being tested. For example, a p-value of 0.9 means 90% of samples taken from the hypothesized distribution have a greater chi-square statistic value. A p-value of 0.01 means only 1% of the samples have a greater chi-square statistic value. In terms of the chi-square test, the hypothesis that the sample comes from the hypothesized distribution is not rejected at a certain confidence level if the p-value is greater than that confidence level. For example, a p-value of 0.03 would result in a rejection of the hypothesis at the 5% confidence level, but not the 1% confidence level. Typically, the confidence level is set to 5% [175].

It is important to note that the p-value does not in any way indicate the probability

that the sample came from the hypothesized distribution [175]. The p-value is a random variable that is calculated using the distribution of a test statistic that has been applied to multiple samples from the hypothesized distribution [114]. Therefore, a sample that comes from the hypothesized distribution can have p-value anywhere between 0 and 1.

The chi-square test is a very popular goodness of fit test because it is simple to implement. However, the test requires binning the data, and the method used to bin the data affects the outcome. In addition, when the number of samples in a bin is small, the test can be inaccurate [120]. Because of this, the Kolmogorov-Smirnov (KS) test is generally preferred [143].

The KS test relies on the empirical CDF, defined in Equation 74 [143].

$$F_e(x) = \begin{cases} 0 & if x \leq x_1 \\ k/n & if x_k \leq x \leq x_{k+1}, \quad for \quad k = 1, ..., n-1 \\ 1 & if x_n \leq x \end{cases} \tag{74}$$

Here, $x_1 \leq x_2 \leq ... \leq x_n$. The KS test works in the same manner as the chi-square test in that a test statistic is calculated using the sample and the hypothesized distribution. A p-value results based on the distribution of that test statistic using samples that come from the hypothesized distribution. The KS test uses a maximum difference between the empirical CDF and the CDF of the hypothesized distribution as the test statistic [120].

$$D = \max_{1 \leq k \leq n} \left(F(x_k) - \frac{k-1}{n}, \frac{k}{n} - F(x_k) \right) \tag{75}$$

Here, $F(x)$ is the CDF of the hypothesized distribution.

In many applications, samples sizes are small and data is hard to generate. In this case, however, the test functions can be sampled thousands of times. Because of this, the process of generating a distribution of sample minima (using random searches) and fitting a GEV distribution will be repeated 100 times for each test function. This

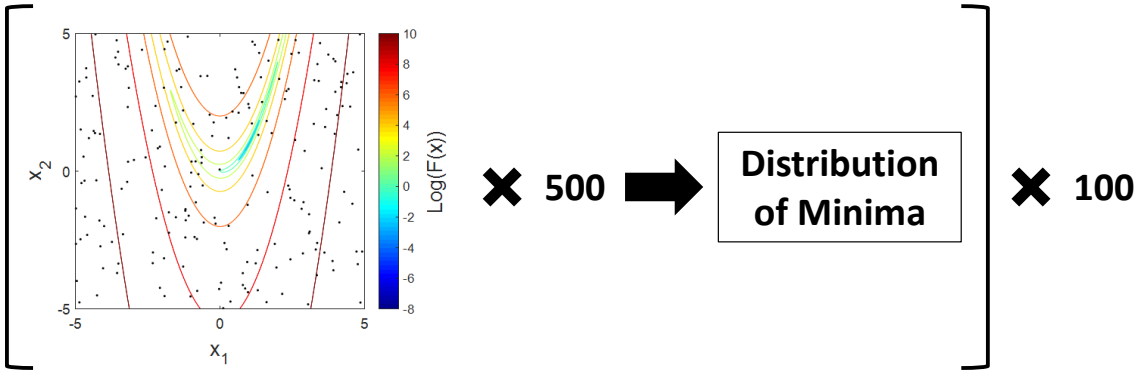


Figure 48: Experimental process for testing the application of EVT to optimization

means the results do not rely on a single p-value. Instead, the percentage of p-values that are above a certain threshold, 0.05 in this case, will be used.

Figure 48 illustrates the process. A random searches is performed on the function in question, resulting in a minimum value for that search. This process is repeated 500 times, based on the study done by Hüsler [12, 80], to result in a distribution of minima. This distribution will be compared to the GEV distribution using the chi-square and KS tests. That entire process is repeated 100 times and the number of times the p-values are above 0.05 is reported. This is done for each of the eight test functions described in the following section.

3.3.1.1 Test Functions

A set of eight test functions is presented here for application in this experiment. The test functions here are chosen from the literature to emulate some of the characteristics of the trajectory optimization problem as well as represent generic optimization problems. These functions were chosen based on Hüsler's work [12, 80], another trajectory optimization study that used test functions [162], and a survey of benchmark functions for optimization problems provided by Jamil [83].

Rosenbrock Function The Rosenbrock function is given in Equation 72 in Section 3.2.4.2. This function is commonly used as a test function for optimization. A contour

plot using a log scale is shown in Figure 49(a).

Function 1 Function 1 is given in Equation 76. This function has a very wide valley where the minimum exists. Figure 49(b) provides a contour plot using a log scale.

$$F(x) = exp(\frac{1}{2}x_1^2 + \frac{1}{2}x_2^2 - \frac{25}{2}) + sin(4x_1 - 3x_2)^2 + \frac{1}{2}(2x_1 + x_2 - 10)^2 \tag{76}$$

Function 2 Function 2 is given in Equation 77. In contrast to Function 1, this function has a very steep drop at the minimum, as seen in Figure 49(c).

$$F(x) = (x_2 - x_1^2)^2 + x_1^2 \tag{77}$$

Alpine 2 The Alpine 2 function is given in Equation 78. This function can be used in any dimension, but in this case n is set to 3. The Alpine 2 function is multimodal, meaning there are a number of local minima. This is representative of the trajectory optimization function, where multiple local optima can exist, especially when using the direct method. Figure 49(d) shows a 3-D plot with $x_3 = 7.91$.

$$F(x) = \prod_{i=1}^{n} \sqrt{x_i} sin(x_i) \tag{78}$$

Helical Valley The Helical Valley function is given in Equation 79. This function has a discontinuity close to the optimum, as seen in Figure 49(e). This plot is created by setting $x_3 = 5$.

$$F(x) = 100\left[(x_2 - 10\theta)^2 + \left(\sqrt{(x_1^2 + x_2^2) - 1}\right)\right] + x_3^2 \tag{79}$$

$$\theta = \begin{cases} \frac{1}{2\pi} tan^{-1}\left(\frac{x_1}{x_2}\right) & if \quad x \geq 0 \\ \frac{1}{2\pi} tan^{-1}\left(\frac{x_1}{x_2} + 0.5\right) & if \quad x \leq 0 \end{cases} \tag{80}$$

Function 3 Function 3 is given in Equation 81. This is a simple function with elliptical contours. This function was used in the work done by Hüsler [12, 80]. A contour plot is shown in Figure 49(f) using a log scale.

$$F(x) = 10x_1^2 + 100x_2^2 \tag{81}$$

Function 4 Function 4 is given in Equation 82. This function is also included in Hüsler's work [12, 80]. There are an infinite number of global minima for this function defined in a circle centered on the origin. These rings can be seen in Figure 49(g).

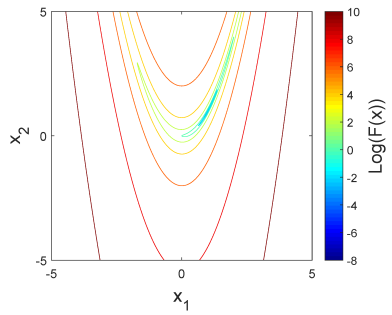
$$F(x) = \frac{sin(x_1^2 + x_2^2)}{x_1^2 + x_2^2} \tag{82}$$

Trid Function The Trid function is given in Equation 83. This function can be used in any dimension, and in this case, n is set to 10. This represents the high dimensionality of the trajectory optimization problem. Figure 49(h) shows a contour plot of this function with all but the first two input variables set to 0.

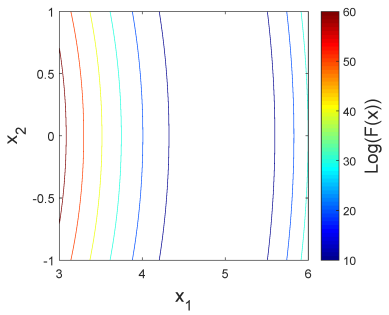
$$F(x) = \sum_{i=1}^{n}(x_i - 1)^2 + \sum_{i=1}^{n} x_i x_{i-1} \tag{83}$$

3.3.2 Experiment Results

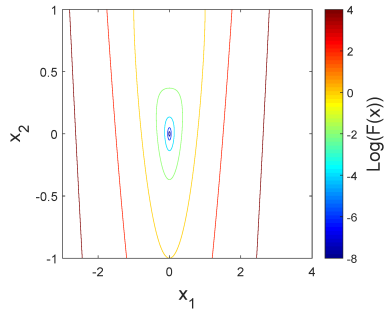
The results from this experiment show that EVT can be applied to optimization in most cases. Table 8 gives the percentage of times the p-value for the corresponding test was over 0.05. For example, the distribution of minimum results from repeated random searches for the Rosenbrock function had a p-value greater than 0.05 96 times out of 100 for the chi-square test and 97 times out of 100 for the KS test when the hypothesized distribution is the GEV distribution. Recall that the p-value is a random variable [114], so this result indicates that the data is indeed distributed as a GEV distribution. The results are similar for all the remaining test functions but one. Function 4 only has a p-value greater than 0.05 56% of the time when the chi-square



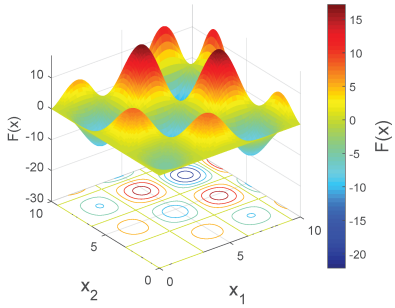
(a) Rosenbrock



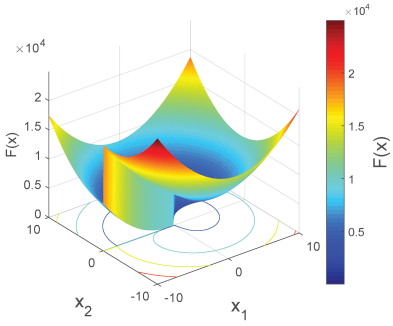
(b) Function 1



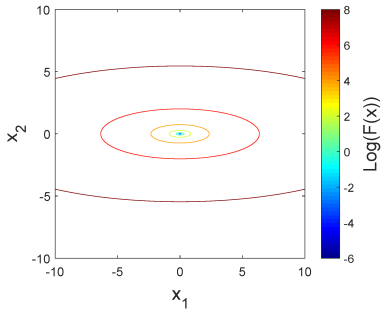
(c) Function 2



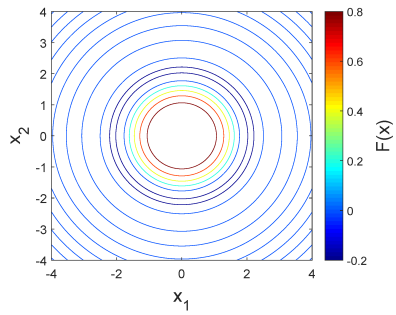
(d) Alpine02, $x_3 = 7.91$



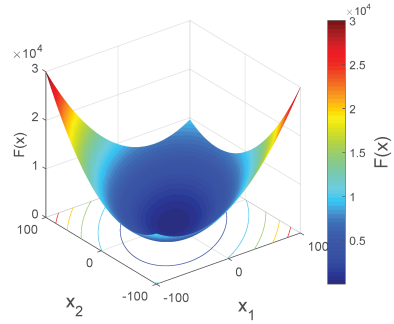
(e) Helical Valley, $x_3 = 5$



(f) Function 3



(g) Function 4



(h) Trid, $x_i = 0 \quad for \quad i = 3, ..., 9$

Figure 49: Test functions for application of EVT to optimization

Table 8: Results of application of EVT to test functions

Test Function	Percentage of p-values ≥ 0.05	
	Chi-square Test	Kolmogorov-Smirnov Test
Rosenbrock	96	97
Function 1	94	96
Function 2	94	95
Alpine 2 3d	100	100
Helical Valley	97	100
Function 3	91	93
Function 4	56	31
Trid 10d	100	100

test is used and 31% of the time when the KS test is used. This is because Function 4 actually has an infinite number of global minima in a ring around the origin, as seen in Figure 49(g). For every other test function, however, the results show that the data is distribution as a GEV distribution. As a comparison, when the data from the Rosenbrock function is tested against a normal distribution, the chi-square and the KS tests both reject the hypothesis every time it was tested.

3.3.3 Answer to Research Question 3

The answer to Research Question 3 is EVT applies to optimization problems by considering a random global search as analogous to a sample from a population and the extremum of that global search as analogous to the sample extremum. This results in a GEV distribution, as predicted by EVT, for all the test functions but one, where an infinite number of global minima existed. Hypothesis 3 is accepted except for the case of an infinite number of global minima. Fortunately, the trajectory problem of interest in this thesis is not characterized by an infinite number of global minima, so EVT can be applied. The next section will discuss how EVT can be applied specifically to the trajectory problem.

3.4 Extreme Value Theory and Trajectory Optimization

This thesis is focused on applying trajectory optimization to enable rapid and accurate launch vehicle performance evaluation in the context of conceptual design. In the previous section, EVT was applied to optimization in general. How can EVT be applied to trajectory optimization? Trajectory optimization can be thought of as finding the trajectory in the population of all possible trajectories that optimizes a value of interest. This was illustrated in Section 3.1.1. In this thesis, the propellant remaining (abbreviated as prop rem in some plots) in orbit will be the objective function and will be referred to as the performance. The performance associated with the optimal trajectory will lie at the tail end of the distribution of performances. EVT provides a way of making inferences about the tail end of distributions. In the section preceding this one, it was shown that EVT can be applied to optimization in the form of repeated random searches. For the trajectory optimization problem, however, random searches are not effective in finding optimal trajectories. In a study for a problem similar to the one described in Section 2.5, a Latin hypercube search, similar to a random search, of over 3 million cases failed to produce a single trajectory that inserted into the target orbit [162]. This study did not even consider maximizing the payload to orbit. A global search alone, like the random search, is not efficient at finding trajectories that reach orbit or at maximizing the mass that reaches orbit. Because of that, this thesis employs a random search in conjunction with a direct method, as seen in Figure 36. The direct method is a local optimization method that is able to fine tune trajectories so that they insert into the target orbit and maximize the mass that is inserted. The effect of adding a local direct method to the

applicability of EVT to this problem is the subject of Research Question 4.

Research Question 4 - Does EVT apply to the launch vehicle optimization problem when the random global search is coupled with a local direct method?

This question can be broken down into two sub-questions that will be addressed through experimentation. Recall from Section 3.2.4 that EVT was concerned with two questions. The first was what kind of distributions did the distribution of sample maxima (or minima) converge to. The answer is the GEV distribution. The second was for what kind of population distributions did the distribution of sample maxima (or minima) converge to the GEV distribution. The population distributions whose distributions of sample maxima (or minima) do converge to the GEV distribution are said to be in the domain of attraction of the GEV distribution. The optimization problem dealt with in this thesis can be thought of in terms of a population distribution, as discussed in Section 3.1.1. This leads to the first of the two sub-questions, Research Question 4.1.

Research Question 4.1 - Is the underlying population of performances for a given vehicle in the domain of attraction of the GEV distribution?

If this question is answered affirmatively, then the maximum values of global searches will result in a GEV distribution. The GEV distribution can then be used to estimate the maximum performance of the vehicle in question. Finding the GEV distribution in this way, however, is computationally expensive because it requires

the performance population. Finding this performance distribution is not a simple task. Instead, local optimization is used in conjunction with the random search, as shown in Figure 36. The local trajectory optimization method takes purposeful steps to increase the performance. Each answer from a trajectory optimization tool will still be in the performance distribution for a vehicle, but it lie at the tail end of the performance distribution, similar to the maximum value of a random sample taken from the performance distribution. The difference is that a trajectory optimization simulation requires much less computational effort. Repeated local optimization will result in a set of performances that lie at the tail end of the distribution. An example of this process was already shown in Figures 38 and 39. However, optimization analyses and sample maxima are not necessarily the same. This leads to the second sub-question for Research Question 4.

Research Question 4.2 - Is the distribution of results from repeated direct optimization analyses for a vehicle analogous to the distribution of sample maxima?

Both these sub-questions will be answered through experiments. These questions represent a significant contribution to the field of EVT as well as trajectory optimization. The analogy between optimized solutions and sample maxima can be very useful. In general it is much easier to run an optimization algorithm a few times than to generate the population distribution that the optimization algorithm will work with. The other contribution relates to the field of aerospace. As far as the author can tell, the performance distribution of a launch vehicle has never been considered. Indeed it is generally not thought of as useful. In the context of EVT, however, it is extremely useful, and may provide insight into the launch vehicle optimization process. Once a GEV distribution is found that describes tail end of the population

distribution, it can be used to estimate the maximum performance. A metric from this distribution not only uses all the data points available, it also is based on the expected form of the data points. Metrics that could be used include the mean or a percentile, such as the 95th percentile. The reason the 100th percentile or 99th percentile is not used is because this particular statistic will be more sensitive to changes in the sample. The mean will not be as sensitive, but is farther from the maximum. A statistic like the 95th percentile provides a consistent measure close to the maximum performance.

The distinction between the control structure generation in Section 2.6 and the performance optimization dealt with in this section is important to understand. The former dealt with how to set up the launch vehicle trajectory optimization problem, specifically how to model the control function. In this section, the optimization task of finding the best values for the control parameters in the control structure selected in Section 2.6 will be addressed via EVT.

3.4.1 Experimental Setup for Research Question 4.1

The fourth experiment in this thesis will address Research Question 4.1. In the previous experiment (Section 3.3), the only test function that did not exhibit behavior predicted by EVT was Function 4, characterized by an infinite number of global minima in a circle around the origin. The trajectory optimization problem does not behave like this, and therefore it is expected that the underlying performance distribution for this problem will be in the domain of attraction of the GEV distribution.

This leads to Hypothesis 4.1.

Hypothesis 4.1 - If the underlying performance population for a given vehicle is used to generate a distribution of sample maxima, that distribution will be distributed as a GEV distribution and therefore the population will be in the domain of attraction of the GEV distribution.

As with the second experiment in Section 2.6.3, the set of 21 representative vehicles will be used to characterize the design space of interest. The distribution of performances can be found for each of the 21 vehicles. These performances represent trajectories that end in the target orbit, but are not optimized to minimize the amount of propellant required to achieve that orbit. An example of this was shown in Figure 37. Once the underlying performance distribution has been found, there are three ways to determine if it lies in the domain of attraction of the GEV distribution. The first and simplest way is to parametrically approximate the underlying distribution for each vehicle. A list of parametric distributions that lie in the domain of attraction of the GEV is provided by Charras [38]. This may prove difficult, as not all distributions are well approximated by a parametric distribution and there is no evidence that the underlying population distributions will be well approximated by parametric distributions. The second way is to generate a distribution of sample maxima directly from the data and compare it to the GEV distribution. A third way is to approximate the underlying distribution with a non-parametric distribution. Non-parametric distributions are very flexible and can be used to approximate virtually any distribution. However, there is no list of non-parametric distributions that fall in the domain of attraction of the GEV distribution. This means that a distribution of sample maxima will have to be generated from the non-parametric

distribution and compared to the GEV distribution. Parametric and non-parametric distributions were reviewed in Section 3.2.

3.4.2 Vehicle Performance Distributions

The vehicle performance distribution represents all the possible paths a vehicle can take to the target orbit. An example of all the paths a vehicle can take to orbit was shown in Figure 37. Certain paths are infeasible because they collide with Earth, they never reach the velocity required, or they burn all available propellant mass. However, there is a set of paths that fly the payload from the launch point to the target orbit. The goal of an optimizer is to find the path that exists in this set of paths that requires the least amount of propellant. Using statistical terminology, the goal of an optimizer is to find the extremal value of the set of paths. The goal of this experiment is to show that EVT can be applied to the vehicle performance distribution. The first step in that endeavor is to find the vehicle performance distributions.

The trajectories found in the vehicle performance distribution are generated by running the industry standard trajectory optimizer POST without optimization. This means that no effort is made to minimize the amount of propellant required to reach the target orbit. Reaching the target orbit, however, is not a trivial task. The tool POST allows for running trajectories without optimization but with orbit targeting. This means each vehicle will fly to the desired orbit, but not necessarily in an optimal way.

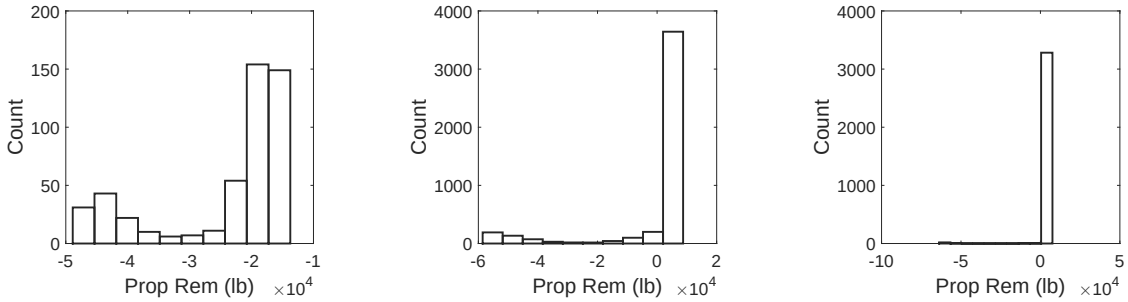
For each vehicle, the underlying vehicle performance distribution was found by running $25,000$ cases without optimization but with targeting. These cases only differ in the initial guess for the control parameters. These were generated randomly from a uniform distribution. The same set of $25,000$ initial guesses were used for each of the 21 vehicles.

The performance results from these cases are shown in Figures 50 and 51, where

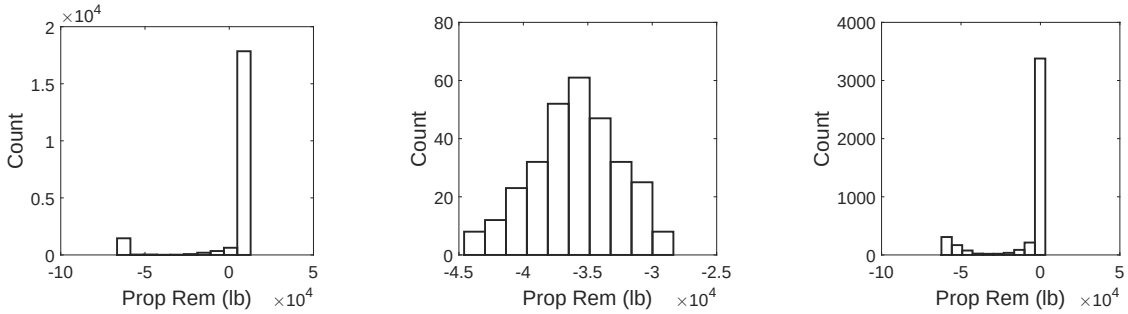
Prop Rem is the propellant remaining. These distributions include a wide range of propellant remaining at the target orbit. Negative propellant remaining indicates this trajectory required more propellant to reach orbit than the baseline vehicle required in Section 2.5. It does not mean that the vehicle reached orbit with negative mass. Certain vehicles, such as Vehicle 1 will always underperform the baseline vehicle because of its design variable values. Other vehicles, like Vehicle 2 will sometimes perform better and sometimes perform worse than the baseline. It is interesting to note that some vehicles include a very wide range of propellant remaining, but most of the trajectories perform in a specific range. Vehicle 3 in Figure 50(c) is an example of this. There is a large spike where most of the trajectories lie, but some performances result in very negative values of propellant remaining. These are very few compared to the over 3000 trajectories included in the single bar. So few, in fact, that they are barely visible in the plot. These trajectories represent statistical outliers.

Statistical outliers are points in a distribution that lie outside a particular range based on the parameters of that distribution. Outliers can occur for several reasons, but for this thesis the outliers simply represent physics-based deviations in the underlying performance distributions. There are many ways to classify outliers [77]. For this thesis, the definition used will be based on the interquartile (IQ) range, which is the distance between the first and third quartiles. A typical definition for an outlier is defined by the range $[Q_1 - 1.5IQ, Q_3 - 1.5IQ]$ [97]. If a data point lies outside this range, it is excluded. For this thesis, only the lower limit will be evaluated to determine outliers. The upper limit represents the points of most interest, where the best performances lie.

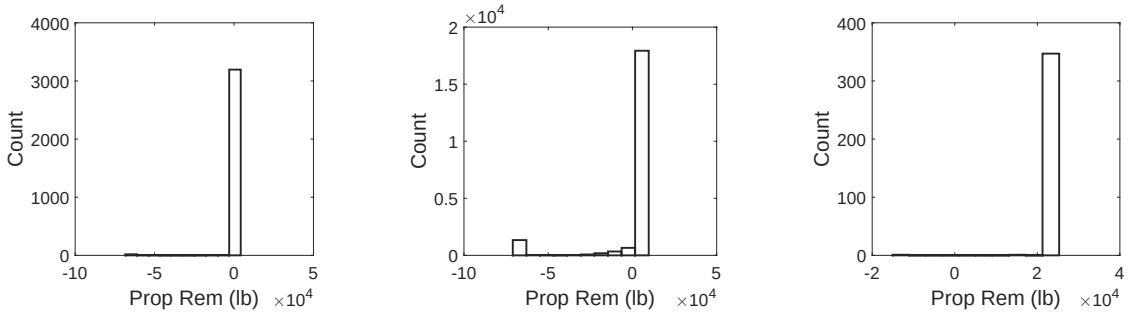
The vehicle performance distributions without outliers, as defined in this thesis, are shown in Figures 52 and 53. The distribution of performances for Vehicle 3 is shown with and without outliers in Figures 50(c) and 52(c) respectively. As explained in the previous paragraph, only outliers below the first quartile are removed. These



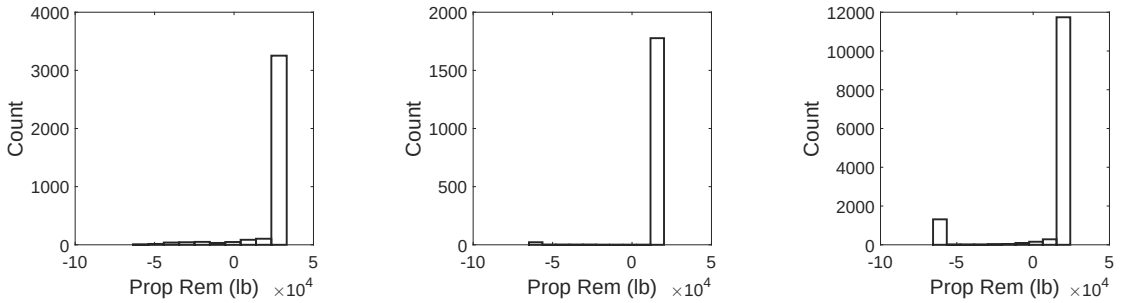
(a) Vehicle 1 (b) Vehicle 2 (c) Vehicle 3



(d) Vehicle 4 (e) Vehicle 5 (f) Vehicle 6

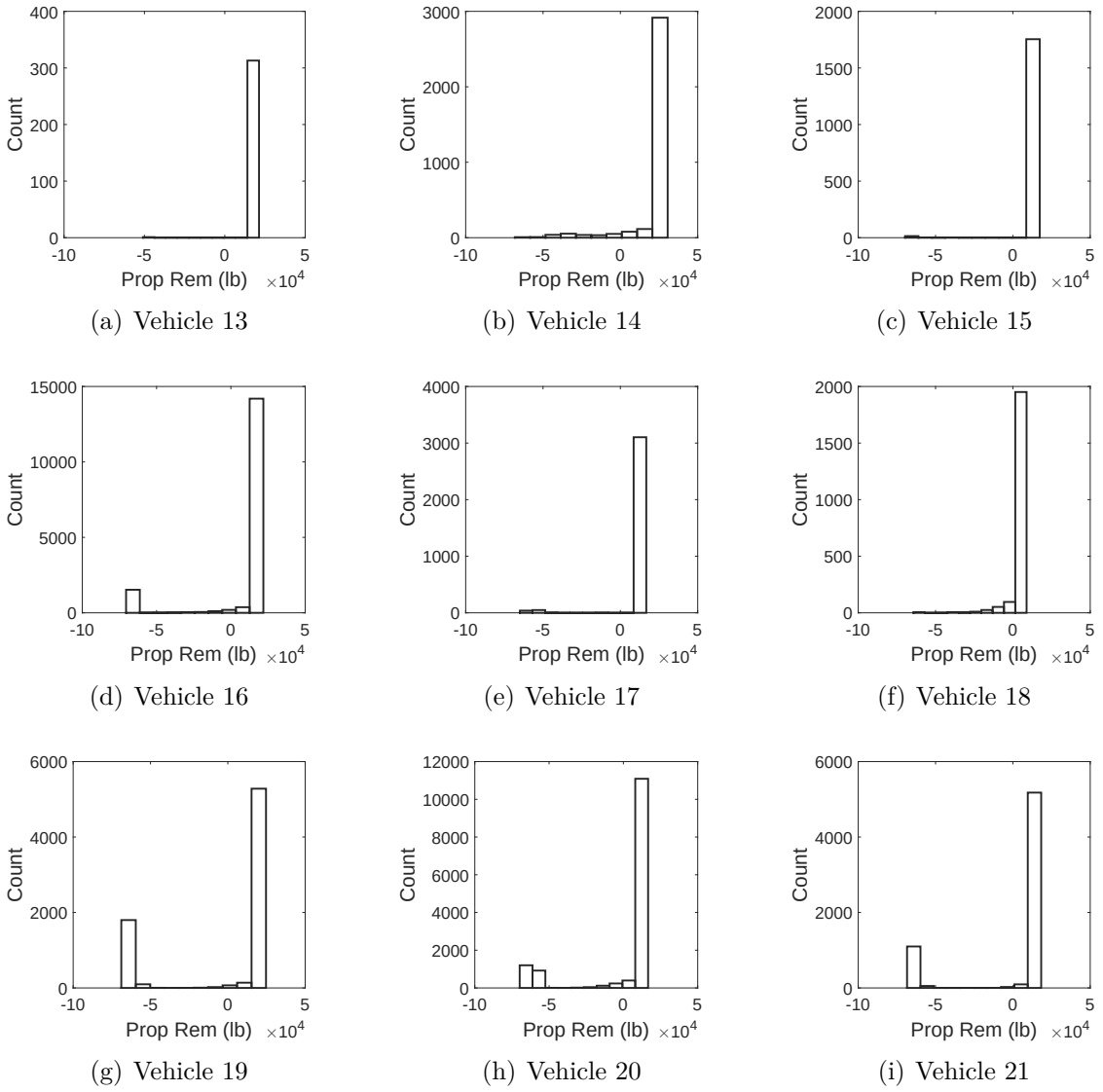


(g) Vehicle 7 (h) Vehicle 8 (i) Vehicle 9



(j) Vehicle 10 (k) Vehicle 11 (l) Vehicle 12

Figure 50: Performance populations for vehicles 1 - 12



(a) Vehicle 13 (b) Vehicle 14 (c) Vehicle 15

(d) Vehicle 16 (e) Vehicle 17 (f) Vehicle 18

(g) Vehicle 19 (h) Vehicle 20 (i) Vehicle 21

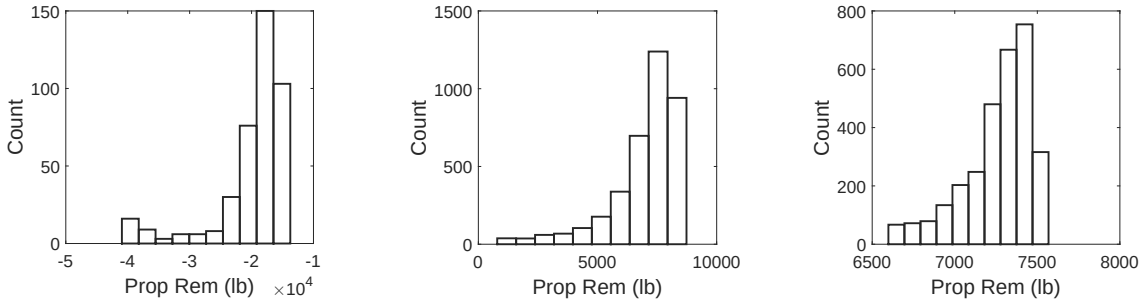
Figure 51: Performance populations for vehicles 13 - 21

distributions represent what an optimizer would have to work with when finding the optimal trajectory. These distributions are by no means exhaustive (it would require an infinite number of repetitions to find the entire distribution) but they are representative of the underlying performance distribution and will be used to determine if EVT is applicable for this problem. For the remainder of this section, the distributions with outliers removed, shown in Figures 52 and 53, will be used for figures. However, it will be shown that the conclusions of the analysis does not depend on whether or not outliers are included.

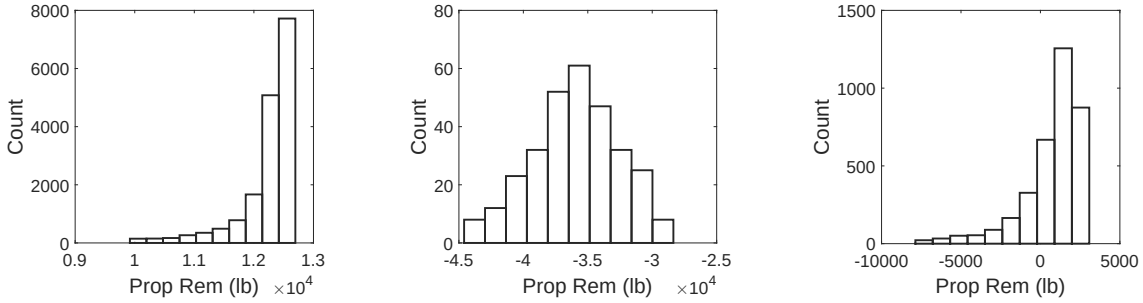
Table 9 gives the number of successful cases out of the $25,000$ run, with and without outliers, for each of the vehicles. It is interesting to note that some vehicles had over 50% of the repetitions return a successful trajectory, while others had less than 10%. This showcases how the optimizer's behavior can be significantly different for vehicles throughout the design space.

3.4.3 Experiment Results

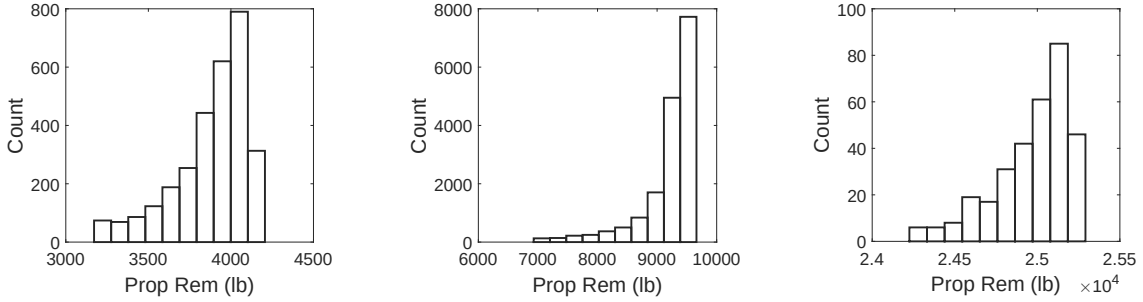
Three options were identified in Section 3.4.1 for characterizing the vehicle performance distributions. Recall the goal is to determine whether these distributions lie in the domain of attraction of the GEV distribution. The first option is to find parametric distributions that fit the empirical distributions. These fits can be compared to a list of parametric distributions that lie in the domain of attraction of the GEV distribution. The second option is to directly sample from the data. If a distribution lies in the domain of attraction of the GEV distribution, then the distribution of sample maxima will converge to the GEV distribution. For this option, a random sample is taken from the data for each vehicle. The maximum of that random sample is then stored. The process can be repeated until a distribution of sample maxima is obtained. This distribution can then be compared to the GEV distribution. The third option is similar to the second, but instead of sampling directly from the data,



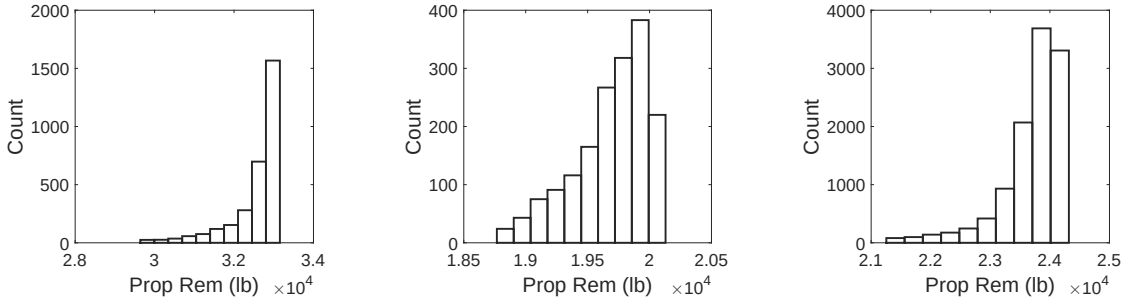
(a) Vehicle 1 (b) Vehicle 2 (c) Vehicle 3



(d) Vehicle 4 (e) Vehicle 5 (f) Vehicle 6

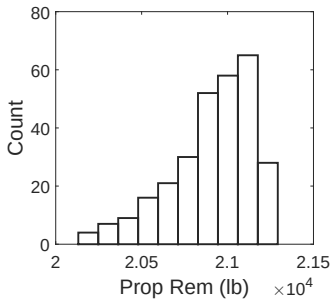


(g) Vehicle 7 (h) Vehicle 8 (i) Vehicle 9

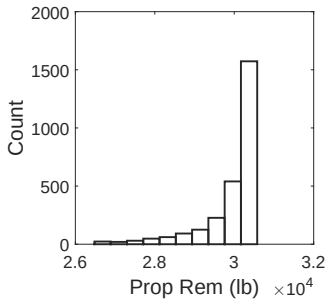


(j) Vehicle 10 (k) Vehicle 11 (l) Vehicle 12

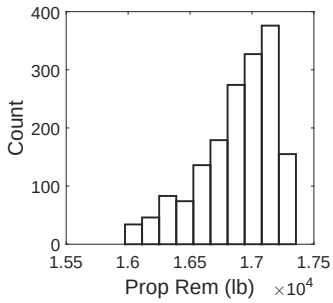
Figure 52: Performance populations without outliers for vehicles 1 - 12



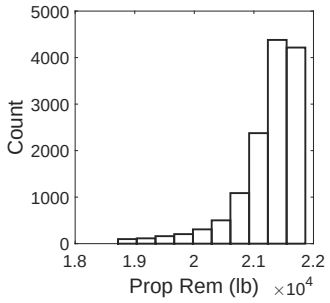
(a) Vehicle 13



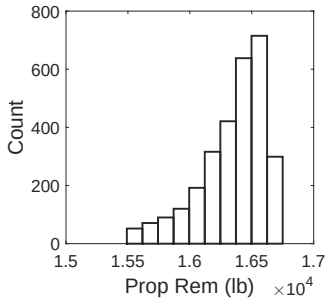
(b) Vehicle 14



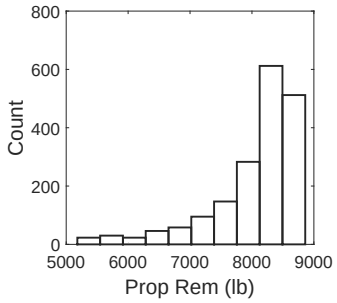
(c) Vehicle 15



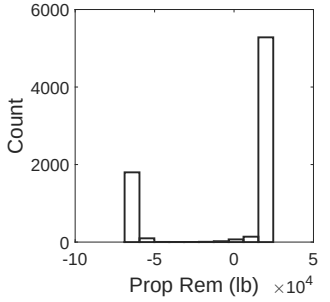
(d) Vehicle 16



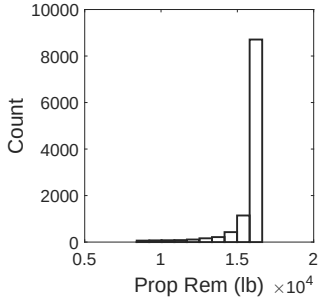
(e) Vehicle 17



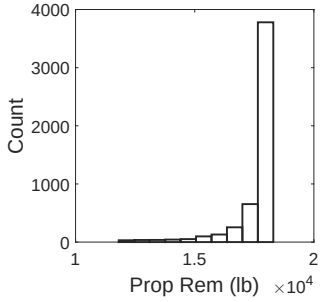
(f) Vehicle 18



(g) Vehicle 19



(h) Vehicle 20



(i) Vehicle 21

Figure 53: Performance populations without outliers for vehicles 13 - 21

Table 9: Successful non-optimized trajectory repetitions for each vehicle

Vehicle	All Points		Outliers Excluded	
	Count	Percent	Count	Percent
1	487	1.9	407	1.6
2	4448	17.8	3699	14.8
3	3308	13.2	3020	12.1
4	20644	82.6	16816	67.3
5	300	1.2	300	1.2
6	4329	17.3	3540	14.2
7	3226	12.9	2960	11.8
8	20610	82.4	16823	67.3
9	349	1.4	321	1.3
10	3677	14.7	3036	12.1
11	1803	7.2	1702	6.8
12	13694	54.8	11153	44.6
13	314	1.3	290	1.2
14	3335	13.3	2739	11.0
15	1770	7.1	1684	6.7
16	16535	66.1	13445	53.8
17	3198	12.8	2914	11.7
18	2149	8.6	1829	7.3
19	7422	29.7	7422	29.7
20	14032	56.1	11064	44.3
21	6451	25.8	5113	20.5

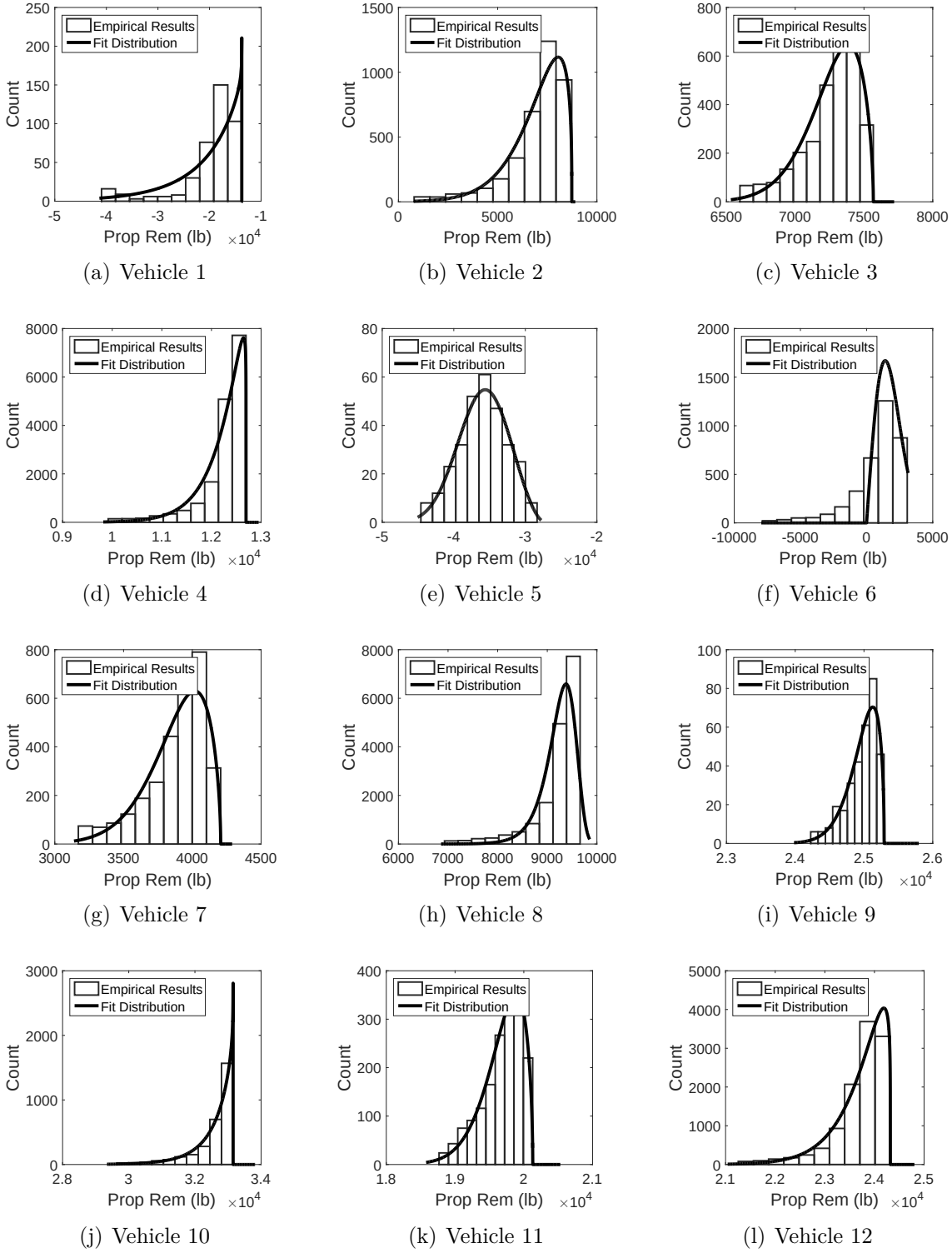
sample from a non-parametric distribution fit to the data. The maxima from random samples taken from this non-parametric fit distribution will result in a distribution that can be compared to the GEV distribution. Each of these three options were performed and are discussed in the following sections.

3.4.3.1 Parametric Distributions of the Vehicle Performance Data

Parametric distributions can be very useful in describing data. Only a handful of parameters, as few as 2, can describe an entire distribution. The challenge with parametric distributions is that not all data sets can be described parametrically. This was explained and shown in Section 3.2.3. For this experiment, a set of parametric distributions will be fit to the data for each vehicle, and the best fit will be chosen. Even the best fit, however, may not be a good fit. The fits will be compared graphically as well as with the two goodness of fit tests described in Section 3.3.1 to assess how well the parametric distribution describes the data.

The histograms for each vehicle with the corresponding parametric fit are shown in Figures 54 and 55. A visual inspection reveals mixed results for the parametric fits. For some vehicles, like vehicle 5, the fit is a good approximation of the data. For other vehicles, however, such as vehicles 19 or 20, the fit is not a good approximation.

The chi-square and KS tests can be performed on each of the parametric fits. The results are shown in Table 10. The p-values for these tests are not included as they for the most part very close to 0. There are 3 vehicles for which the parametric fit is a good approximation according to the chi-square test. The KS test indicates that an additional vehicle has a good parametric fit. It is concluded, therefore, that parametric fits for the performance population distributions are not appropriate and will not be used to determine if the distributions are in the domain of attraction of the GEV distribution.



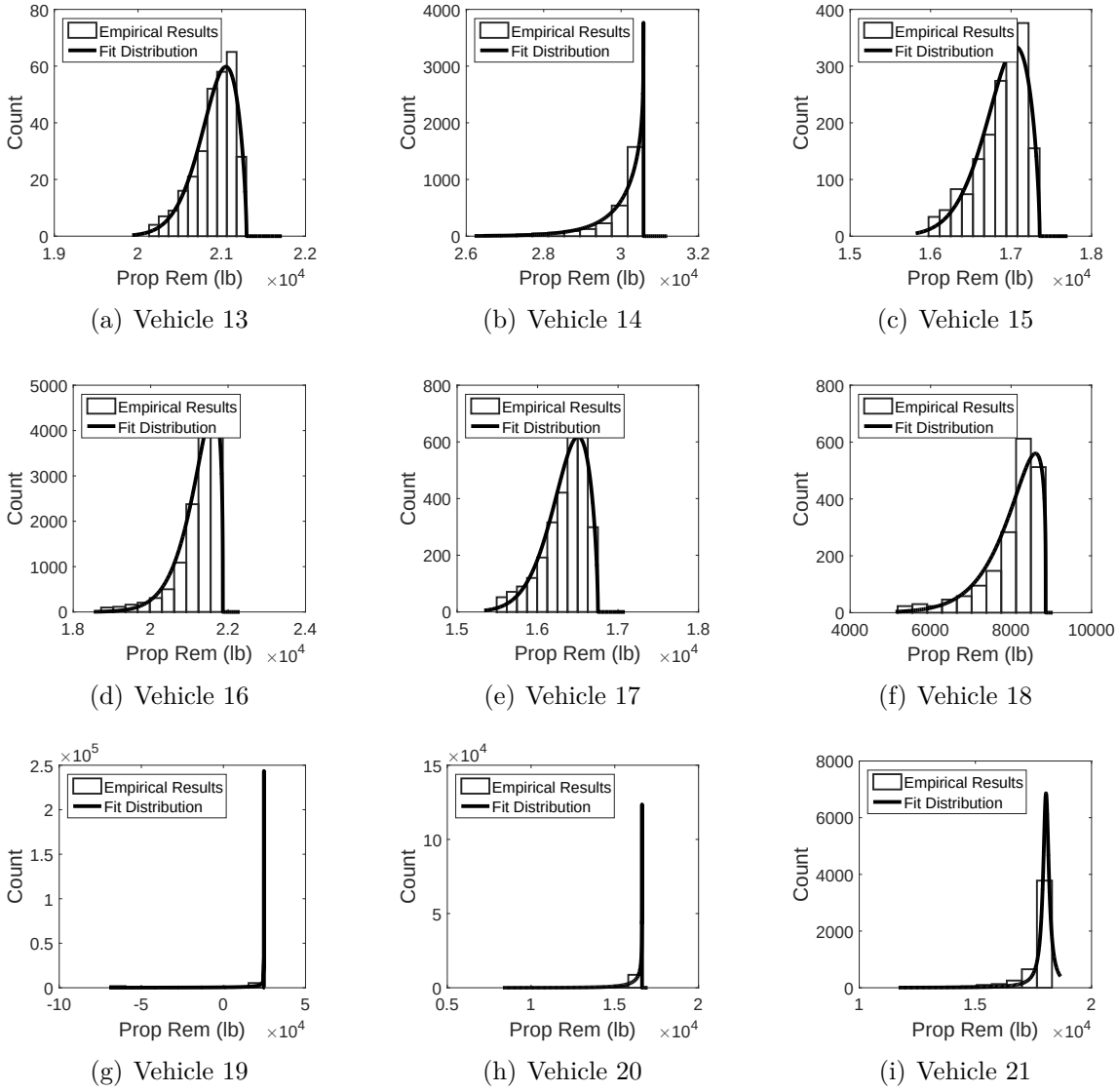
(a) Vehicle 1 (b) Vehicle 2 (c) Vehicle 3

(d) Vehicle 4 (e) Vehicle 5 (f) Vehicle 6

(g) Vehicle 7 (h) Vehicle 8 (i) Vehicle 9

(j) Vehicle 10 (k) Vehicle 11 (l) Vehicle 12

Figure 54: Performance populations with parametric fits for vehicles 1 - 12



(a) Vehicle 13 (b) Vehicle 14 (c) Vehicle 15

(d) Vehicle 16 (e) Vehicle 17 (f) Vehicle 18

(g) Vehicle 19 (h) Vehicle 20 (i) Vehicle 21

Figure 55: Performance populations with parametric fits for vehicles 13 - 21

Table 10: Goodness of fit test results for parametric fits of vehicle performance populations

Vehicle	Chi-square Test	KS Test
1	1	1
2	1	1
3	1	1
4	1	1
5	0	0
6	1	1
7	1	1
8	1	1
9	0	0
10	1	1
11	1	0
12	1	1
13	0	0
14	1	1
15	1	1
16	1	1
17	1	1
18	1	1
19	1	1
20	1	1
21	1	1

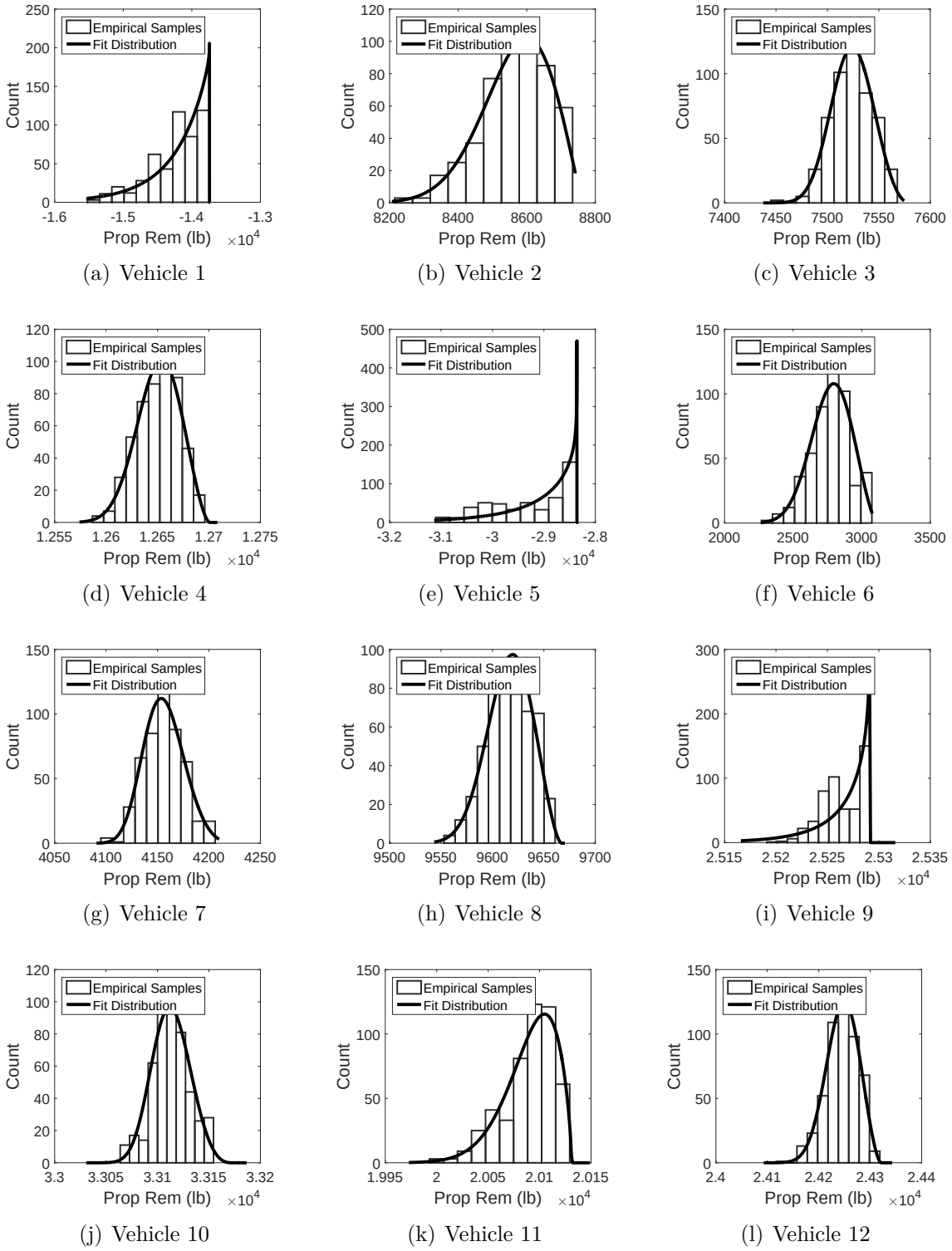
3.4.3.2 Direct Sampling from the Vehicle Performance Data

A second option to test if the underlying distributions, shown in Section 3.4.2, are in the domain of attraction of the GEV distribution is to directly sample from the data. This can be done by randomly generating a sample from the data and identifying the maximum from that sample. This can be repeated until a distribution of sample maxima is obtained. This distribution can be compared to the GEV distribution.

The samples obtained directly from the population performance data each contained 50 items, and a total of 500 samples were taken, resulting in a distribution of sample maxima with 500 items. Figures 56 and 57 show the results for the data as well as the fit GEV distributions for each of the vehicles. A visual inspection reveals that the distributions are a good approximation of the data for most of the vehicles. However, some, like Vehicles 5, 9, or 13 do not pass a visual inspection.

The goodness of fit tests used previously can be used to evaluate the fits. Table 11 shows the results. The p-values are not included because many of them were very near 0. The chi-square test shows that about half of the vehicles fail. Similarly, about half of the vehicles pass the KS test.

It should be noted that this method for testing the underlying distributions relies heavily on the data collected. For some vehicles, only a few hundred data samples are available. This means the maximum values are going to be very similar. This results in an artificial discretization of the distribution of sample maxima. Consider a situation where the population is the set of numbers $\{1, 2, 3, 4, 5\}$. The sample maxima is going to be 4 or 5 in almost every situation. It will never be 4.5, so the distribution of sample maxima will be discretized. For the trajectory problem, the populations contain more than 5 items, but the effect may still be present. A better method that does not suffer from this problem is discussed in the next section.



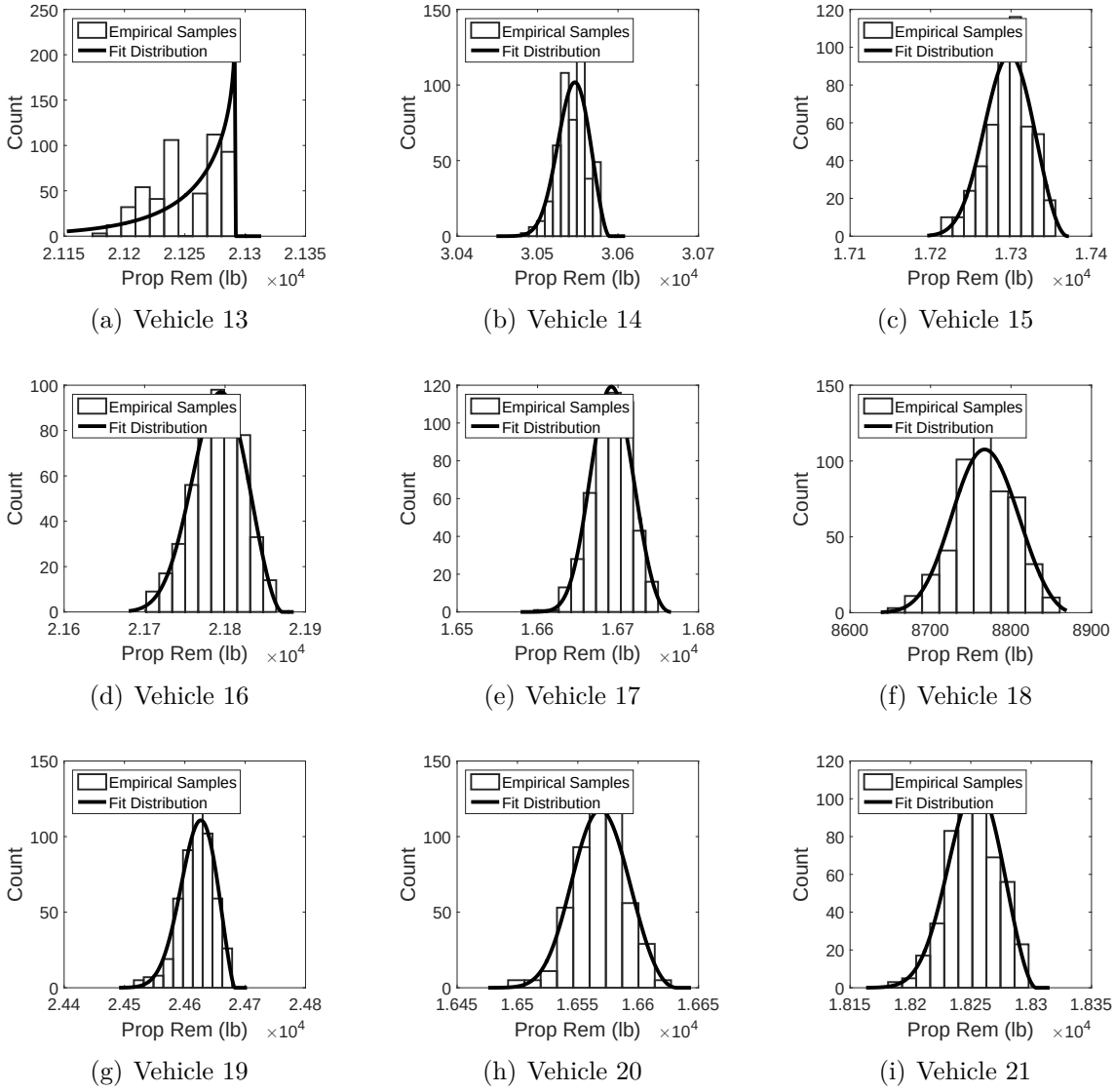
(a) Vehicle 1 (b) Vehicle 2 (c) Vehicle 3

(d) Vehicle 4 (e) Vehicle 5 (f) Vehicle 6

(g) Vehicle 7 (h) Vehicle 8 (i) Vehicle 9

(j) Vehicle 10 (k) Vehicle 11 (l) Vehicle 12

Figure 56: Distribution of sample maxima with direct sampling for vehicles 1 - 12



(a) Vehicle 13 (b) Vehicle 14 (c) Vehicle 15

(d) Vehicle 16 (e) Vehicle 17 (f) Vehicle 18

(g) Vehicle 19 (h) Vehicle 20 (i) Vehicle 21

Figure 57: Distribution of sample maxima with direct sampling for vehicles 13 - 21

Table 11: Goodness of fit test results for GEV distributions using direct sampling of vehicle performance populations

Vehicle	Chi-square Test	KS Test
1	1	1
2	0	1
3	0	1
4	0	0
5	1	1
6	1	1
7	0	0
8	0	0
9	1	1
10	1	0
11	1	1
12	0	0
13	1	1
14	1	1
15	1	1
16	0	0
17	0	1
18	0	1
19	1	0
20	0	0
21	0	0

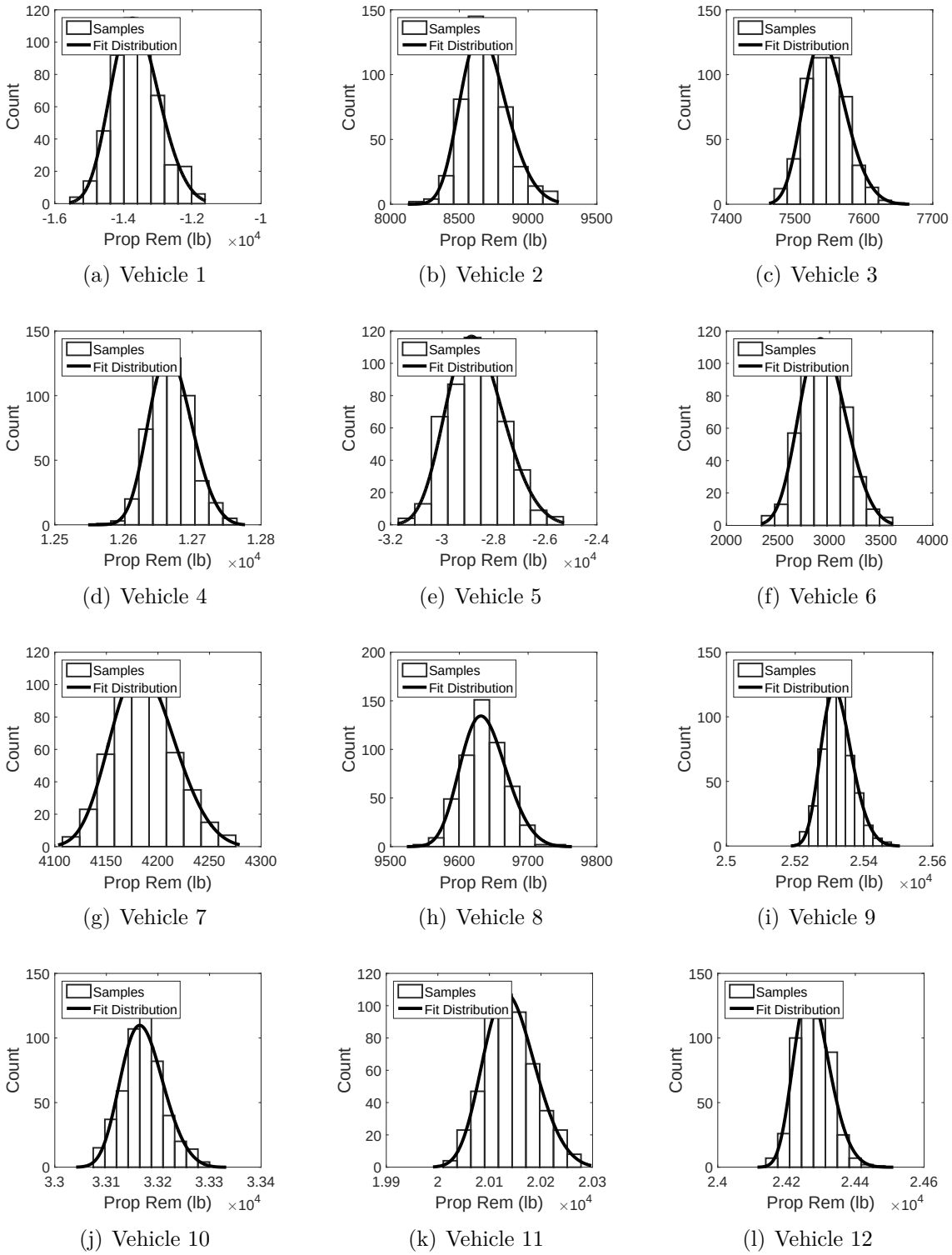
3.4.3.3 Nonparametric Distributions of the Vehicle Performance Data

There are challenges with both parametric fits and direct sampling that yield these methods inappropriate for characterizing the performance distributions of the vehicles. A third method involving nonparametric distributions is presented here. Recall from Section 3.2.2 that nonparametric distributions can be used to fit any set of data and do not rely on a specific form of distribution. A nonparametric distribution provides a way of accurately representing the vehicle performance population data in a continuous form. The fact that it is continuous means it will not suffer from the artificial discretization that the direct sampling did. In this case, kernel density estimates were used to generate the nonparametric distributions. Figures 90 and 91 in Appendix D Section D.1 show the kernel density estimates overlaid on the vehicle performance populations.

Samples can be taken from the nonparametric distribution, and the maximum of each sample recorded to construct a distribution of sample maxima. This distribution can then be compared to the GEV distribution. As in the previous section, the sample size is set to 50 and 500 samples are taken, resulting in a distribution of sample maxima of size 500. Figures 58 and 59 below show the resulting distributions of sample maxima and the GEV distributions fit to the data. A visual inspection reveals that all the GEV distributions approximate the distribution of sample maxima well.

The goodness of fit tests used the previous sections can be applied here to assess the quality of the fits. Table 12 gives the results, including p-values. Both the chi-square and KS tests indicate that the distribution of sample maxima does come from a GEV distribution for each of the vehicles. Recall that the p-values are not an indication of the probability that the data comes from a GEV distribution. Instead, they should be randomly distributed, as is seen in the results.

In Section 3.4.2, the outliers were excluded from the vehicle performance populations in part to aid in finding parametric approximations and in part to result in



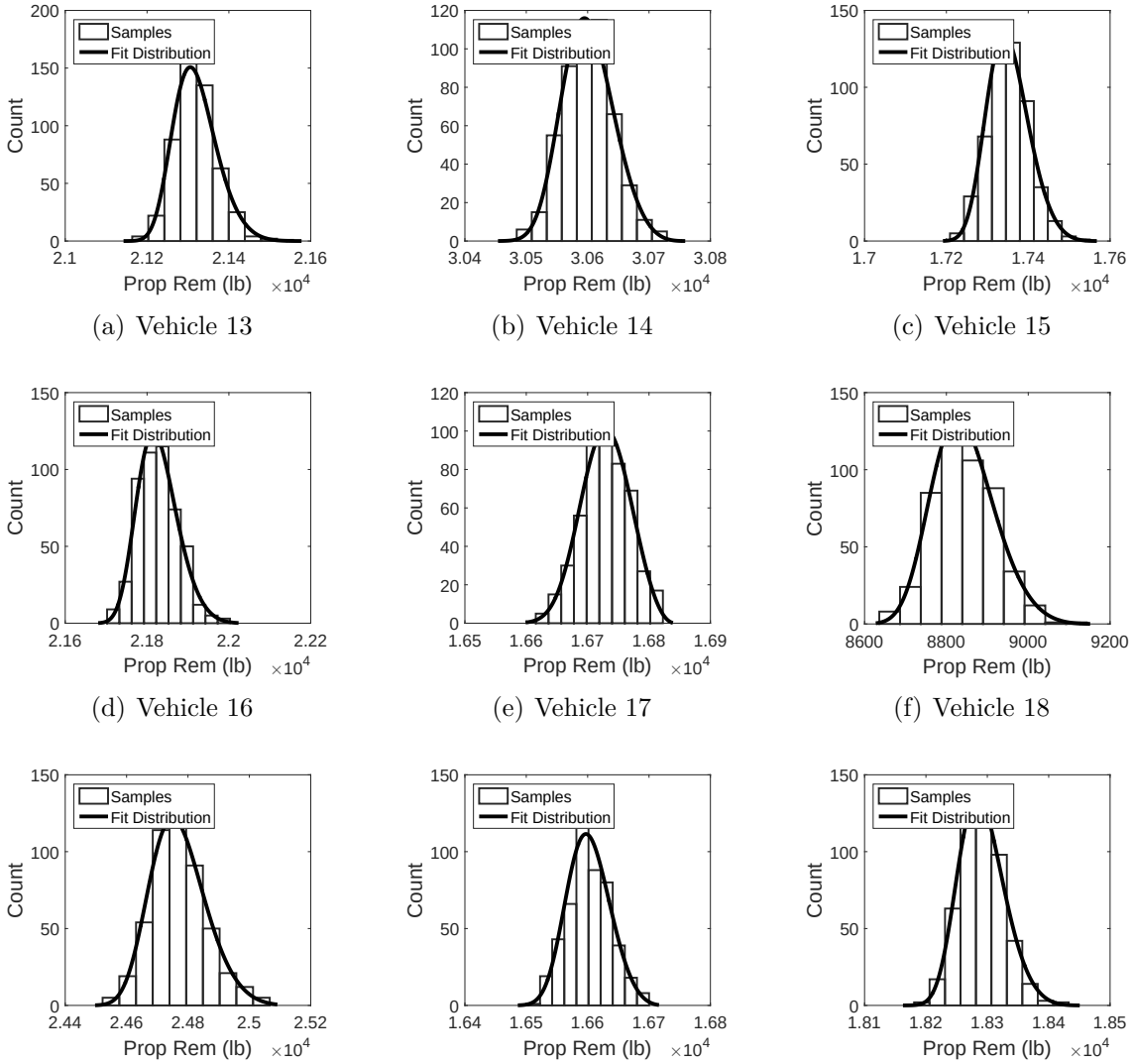
(a) Vehicle 1 (b) Vehicle 2 (c) Vehicle 3

(d) Vehicle 4 (e) Vehicle 5 (f) Vehicle 6

(g) Vehicle 7 (h) Vehicle 8 (i) Vehicle 9

(j) Vehicle 10 (k) Vehicle 11 (l) Vehicle 12

Figure 58: Distribution of sample maxima with nonparametric population estimates for vehicles 1 - 12



(a) Vehicle 13 (b) Vehicle 14 (c) Vehicle 15

(d) Vehicle 16 (e) Vehicle 17 (f) Vehicle 18

(g) Vehicle 19 (h) Vehicle 20 (i) Vehicle 21

Figure 59: Distribution of sample maxima with nonparametric population estimates for vehicles 13 - 21

Table 12: Goodness of fit test results for GEV distributions using nonparametric estimation of vehicle performance populations without outliers

Vehicle	Chi-square Test	p-value	KS Test	p-value
1	0	0.307	0	0.907
2	0	0.154	0	0.504
3	0	0.539	0	0.896
4	0	0.524	0	0.781
5	0	0.603	0	0.970
6	0	0.919	0	0.996
7	0	0.922	0	0.973
8	0	0.462	0	0.501
9	0	0.491	0	0.552
10	0	0.140	0	0.802
11	0	0.943	0	0.997
12	0	0.159	0	0.790
13	0	0.545	0	0.576
14	0	0.779	0	0.920
15	0	0.513	0	0.595
16	0	0.426	0	0.912
17	0	0.706	0	0.793
18	0	0.464	0	0.911
19	0	0.660	0	0.506
20	0	0.054	0	0.493
21	0	0.780	0	0.815

Table 13: Goodness of fit test results for GEV distributions using nonparametric estimation of vehicle performance populations with outliers

Vehicle	Chi-square Test	p-value	KS Test	p-value
1	0	0.575	0	0.770
2	0	0.857	0	0.825
3	0	0.456	0	0.894
4	0	0.715	0	0.998
5	0	0.603	0	0.970
6	0	0.606	0	0.506
7	0	0.223	0	0.595
8	0	0.582	0	0.860
9	0	0.717	0	0.875
10	0	0.680	0	0.902
11	0	0.072	0	0.808
12	0	0.230	0	0.686
13	0	0.612	0	0.808
14	0	0.631	0	0.927
15	0	0.955	0	0.829
16	0	0.055	0	0.199
17	0	0.966	0	0.947
18	0	0.549	0	0.322
19	0	0.660	0	0.506
20	0	0.771	0	0.991
21	0	0.969	0	0.987

better figures for the distributions. However, this is not necessary for in the nonparametric case. Table 13 shows the goodness of fit test results for this process when the outliers are included. Figures are not included as there is no visual difference when outliers are included or excluded. The p-values are different, as expected, but both the chi-square and KS tests indicated that the distribution of sample maxima does come from a GEV distribution. This is not an unexpected result, as the inclusion of the lower outliers will only affect the lower end of the nonparametric distribution. Even if data is selected from that region in a data sample, it will not be the sample maxima. There is an extremely low probability items from the part of the performance distribution will be included in the distribution of sample maxima.

These results show that the performance distributions of the representative vehicle

set tested do indeed lie in the domain of attraction of the GEV distribution.

3.4.4 Answer to Research Question 4.1

The answer to Research Question 4.1 is that the underlying population of performances for a given vehicles does lie in the domain of attraction of the GEV distribution. The experiment showed that this was the case for each of the vehicles in the set of representative vehicles. As predicted in Hypothesis 4.1, the distribution of sample maxima that resulted from the underlying performance population for each vehicle, estimated by a nonparametric distribution, is distributed as the GEV distribution. Because these vehicles are representative of the design space, it is concluded that the population of performances for a given vehicle is also in the domain of attraction of the GEV distribution. In conjunction with the answer to Research Question 3, this means that the maxima of repeated random searches will be distributed as a GEV distribution. However, random searches have been shown to be ineffective in the context of trajectory optimization. The following section will address this problem.

3.4.5 Direct Method Optimization and Extreme Value Theory

Finding the performance population for a vehicle is an expensive task. Thousands of repetitions for each vehicle must be run. Each repetition will require some kind of targeting to reach the final orbit. EVT can be applied to a population by taking repeated samples and estimating the distribution of sample maxima. To apply directly, this would mean finding the performance distribution for each vehicle, which is not desirable.

The answers from Research Questions 3 and 4.1 mean that the maxima of repeated random searches will result in a GEV distribution for the trajectory optimization problem. However, random searches are not a feasible option for solving launch vehicle trajectory problems. As discussed in previously in this section, another method can be applied by drawing an analogy between an optimized result and a sample

maximum. When a random sample is taken, random performances are taken from the performance population. If enough are taken, the maximum performance in the sample will be in the upper tail of the distribution. Similarly, an optimization process starts somewhere in the performance distribution and takes intentional steps toward the upper edges of the distribution. The same result can be arrived at using a sample or optimization. The benefit of using optimization is that the population of performances is not required. This eliminates the need for evaluating thousands of cases required to estimate said distribution. Therefore, instead of using a random search alone, the random search is coupled with a local direct optimization method. This led to the second sub-question, Research Question 4.2, repeated below.

Research Question 4.2 - Is the distribution of results from repeated direct optimization analyses for a vehicle analogous to the distribution of sample maxima?

It is expected that the distribution of optimization results will behave similarly to the distribution of sample maxima, leading to Hypothesis 4.2.

Hypothesis 4.2 - If a random search is used to initialize repeated direct optimization processes for a trajectory problem, then the results will be analogous to a distribution of sample maxima and distributed as a GEV distribution.

As a statistical method, EVT requires that the samples used to generate the distribution of sample maxima be dependent and identically distributed. Using a direct optimizer violates this assumption. Therefore the optimization results may

not be exactly distributed as a GEV distribution. If this is the case, however, the differences between these can be quantified. The following section will discuss the method for testing the hypothesis.

3.4.6 Experimental Setup for Research Question 4.2

Hypothesis 4.2 will be tested using the same set of representative vehicles used in Sections 2.6.3 and 3.4.1. A set of optimization simulations will be performed for each vehicle. These simulations will only differ in the initial guess for the control parameters. The initial guess will be initialized using a global random search method, and will therefore be uniformly distributed. This process represents the coupling of a random search with repeated direct method optimization. For each vehicle, a set of $25,000$ initial guesses will be run. These initial guesses are the same as those used in Section 3.4.1 to test Research Question 4.1. In that case, no optimization was used. Here, the optimizer will be used to maximize the amount of payload to orbit. The results are reported in terms of the amount of propellant remaining, which is offset from the total mass in orbit by a fixed amount.

The result will be a distribution of optimized results for each of the vehicles. This distribution can be compared to a GEV distribution and the goodness of fit tests used previously can be applied. Additionally, the Kullback-Leibler (KL) divergence [94] will also be measured. The KL divergence quantifies the dissimilarity between two distributions. It is not a metric in that the KL divergence between distribution A and distribution B is not equal to the divergence between B and A [87].

The reason the KL divergence is also used in this experiment is because of the nature of the goodness of fit tests, such as the chi-square test and the KS test. These tests rely on a null hypothesis, which states that a set of data comes from a certain distribution. The alternative hypothesis states that the data does not come from that distribution. The result is binary; the null hypothesis is either rejected or it

is not. Recall that p-values, which are used to perform the goodness of fit tests at certain levels of confidence, are not an indication of the probability that a set of data comes from a distribution (see Section 3.3.1). The goodness of fit tests are useful in telling if a set of data comes from a certain distribution, but if the null hypothesis is rejected, these tests do not say anything about how far off the data is from a certain distribution.

Consider an analogy of testing whether or not a pair of two-dimensional lines that pass through the origin are parallel. The goodness of fit tests can be compared to going towards infinity and measuring the distance between the lines. If there is a distance, the null hypothesis is rejected. The KL divergence, on the other hand, is analogous to considering the difference between the two lines over a certain range, and returning a measure of that distance.

As an example, a set of is taken from a normal distribution with mean 0 and standard deviation of 1. Another set of data is taken from a uniform distribution, between -2.5 and 2.5. Both these sets of data are then compared to a hypothesized normal distribution with mean 0 and standard deviation 1.1. Both the chi-square test and the KS test reject the hypothesis that the data comes from the hypothesized distribution. However, the KL divergence for the normal data is 0.010 while for the uniform data it is 0.26, over an order of magnitude higher. The utility of the KL divergence will be shown later in this experiment.

The KL divergence is one example of many (over 60) of a distance or similarity measure between probability density functions [37]. Out of all the measures available, the KL divergence is chosen for two reasons. First of all, only a single measure is needed. This study is not designed to compare different distance metrics and there is not a need to consider more than one. In fact, the need for the large number of similar types of measures has been questioned in the literature [144]. Secondly, the KL divergence is the most commonly used divergence measure in practical studies

[44]. This may be because it was one of the first divergence measures proposed [94], or because it has been shown to provide accurate divergence measures in many situations [44]. In either case, it is implemented in this thesis as the divergence measure.

3.4.7 Experiment Results

The $25,000$ initial guesses used for each vehicle in this experiment did not all result in successful trajectories. Table 14 gives the number of successful cases for each of the vehicles. The number of successful cases when the trajectory is optimized should be equal to or less than the number of successful cases when no optimization is used, as was shown in Table 9. This is not the case for some of the vehicles because additional initial guesses were used to ensure enough optimized repetitions were found to fit the GEV distribution.

Table 14: Successful optimized trajectory repetitions for each vehicle

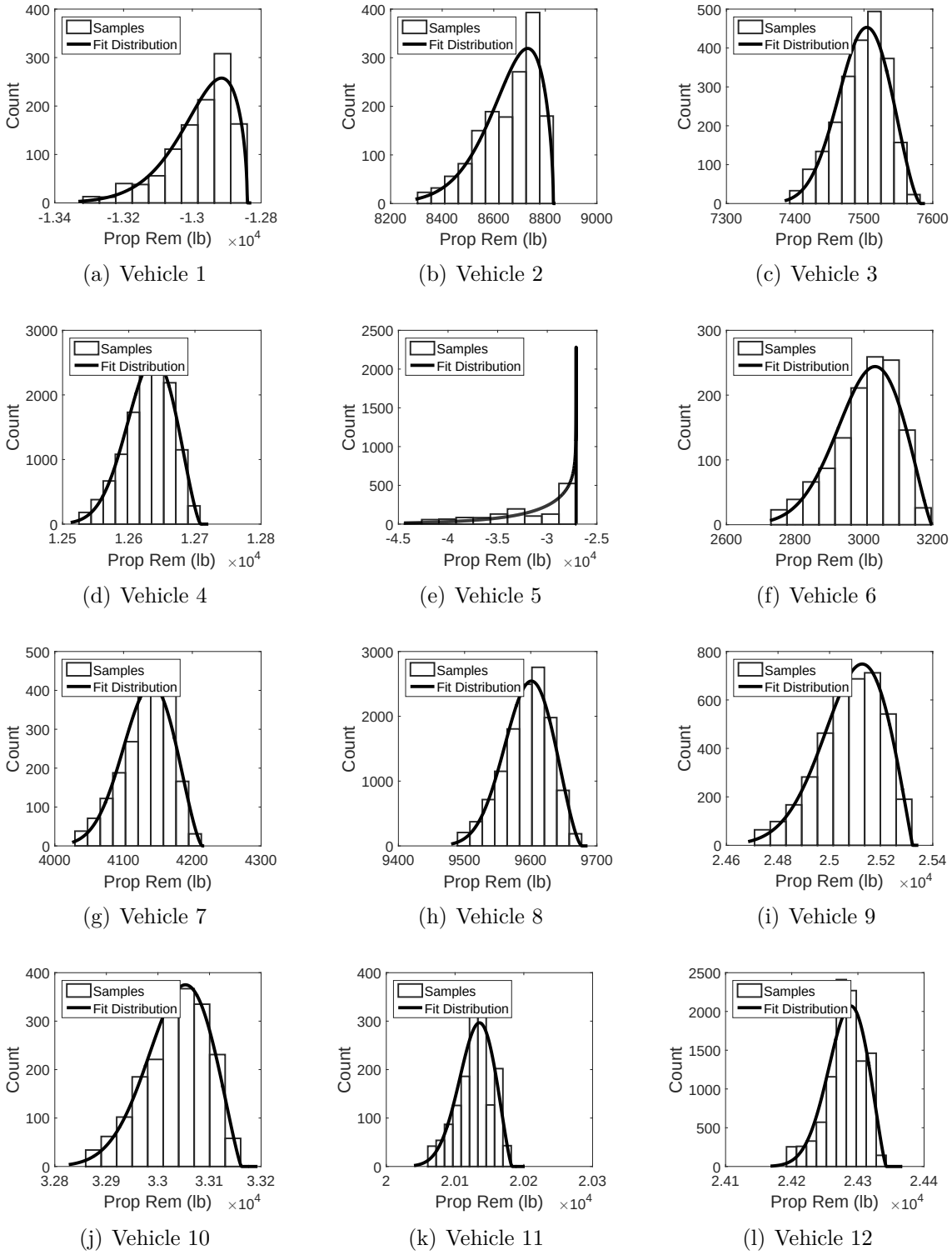
Vehicle	Count	Percent
1	1115	1.5
2	1554	6.2
3	2258	9.0
4	12776	51.1
5	1390	0.6
6	1245	5.0
7	2100	8.4
8	12525	50.1
9	3854	9.6
10	1952	7.8
11	1573	6.3
12	10215	40.9
13	3740	9.4
14	1790	7.2
15	1495	6.0
16	2821	11.3
17	11145	27.9
18	6668	16.7
19	5200	13.0
20	2032	8.1
21	11028	27.6

The performance results of the random search with repeated direct method optimization analyses are shown in Figures 60 and 61, where Prop Rem is the propellant remaining. A GEV distribution is fit to the results for each of the vehicles and plotted alongside the data. A visual inspection reveals that the GEV distribution captures the data well for all but one of the vehicles, Vehicle 5. Vehicle 5 will be excluded from the analysis for reasons explained later on in Section 3.6.1. For now, however, it will still be considered.

The goodness of fit tests used in this thesis can be applied to these data sets and the corresponding GEV distributions. Table 15 gives the results for the chi-square and KS tests. The hypothesis that the data comes from a GEV distribution is rejected for all but two of the vehicles by the chi-square test and for all but five by the KS test. This indicates that sample maxima are not exactly like optimized cases. However, this data alone does not tell the whole story.

Another visual way to compare distribution is to plot the quantiles of the hypothesized distribution against the empirical quantiles from the data. This is known as a Q-Q plot. Figures 62 and 63 show the Q-Q plots for each of the vehicles. The dashed reference line represents what an exact fit would look like. A quartile line is included, but is generally not visible because the data points are overlaid. It is clear from this representation of the data that the GEV distribution does not exactly describe the data. Towards the lower end of the distributions there is a discrepancy between the empirical data and hypothesized GEV distribution. The GEV distribution is predicting that the data at the lower ends of the distribution should be spread even lower. However, the actual values are grouped toward the mean of the data. This is not necessarily an unexpected result. The data was generated using an optimization process, pushing each of the points to the middle and upper ends of the distribution.

The discrepancies between the GEV distribution and the data seems to exist at the lower end of the distribution. At the upper end, the data follows the predicted



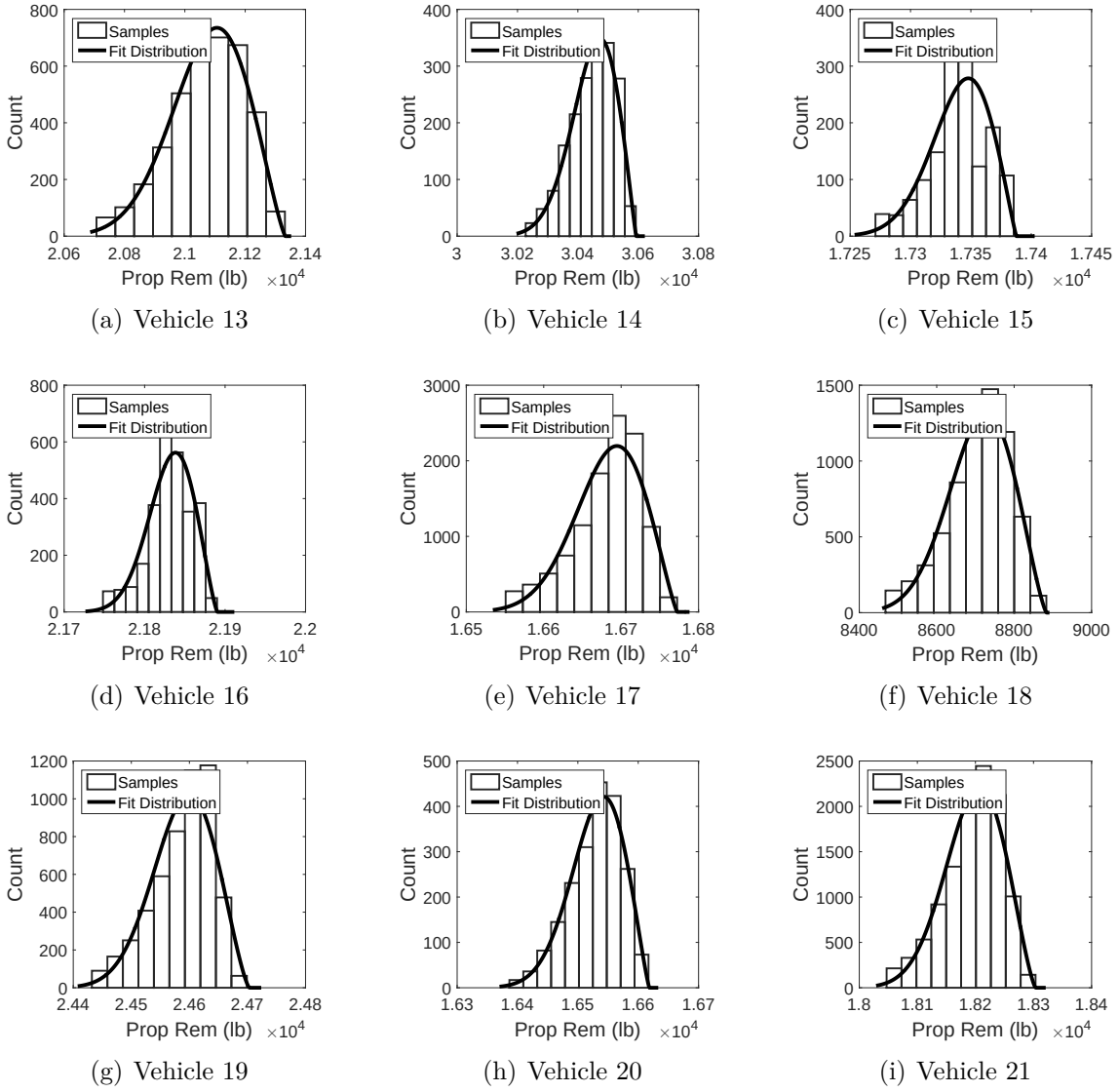
(a) Vehicle 1 (b) Vehicle 2 (c) Vehicle 3

(d) Vehicle 4 (e) Vehicle 5 (f) Vehicle 6

(g) Vehicle 7 (h) Vehicle 8 (i) Vehicle 9

(j) Vehicle 10 (k) Vehicle 11 (l) Vehicle 12

Figure 60: Distribution of repeated optimization analysis for vehicles 1 - 12



(a) Vehicle 13 (b) Vehicle 14 (c) Vehicle 15

(d) Vehicle 16 (e) Vehicle 17 (f) Vehicle 18

(g) Vehicle 19 (h) Vehicle 20 (i) Vehicle 21

Figure 61: Distribution of repeated optimization analysis for vehicles 13 - 21

Table 15: Goodness of fit test results for repeated optimization analysis

Vehicle	Chi-square Test	p-value	KS Test	p-value
1	1	1.09E-04	1	4.28E-02
2	1	3.28E-09	1	1.19E-03
3	1	1.93E-07	1	1.80E-02
4	1	6.63E-19	1	1.11E-04
5	1	6.05E-05	1	2.59E-29
6	1	1.01E-04	1	4.85E-02
7	1	2.72E-06	1	3.32E-02
8	1	2.32E-29	1	7.57E-07
9	0	4.20E-01	0	6.08E-01
10	0	9.50E-02	0	3.95E-01
11	1	2.27E-23	1	3.72E-07
12	1	3.31E-184	1	1.51E-23
13	1	4.17E-03	0	4.62E-01
14	1	3.84E-04	0	1.46E-01
15	1	3.16E-28	1	2.14E-09
16	1	2.01E-42	1	3.75E-09
17	1	3.29E-117	1	1.42E-26
18	1	1.06E-25	1	1.68E-07
19	1	8.79E-60	1	3.29E-16
20	1	2.91E-02	0	8.50E-02
21	1	5.01E-59	1	9.64E-13

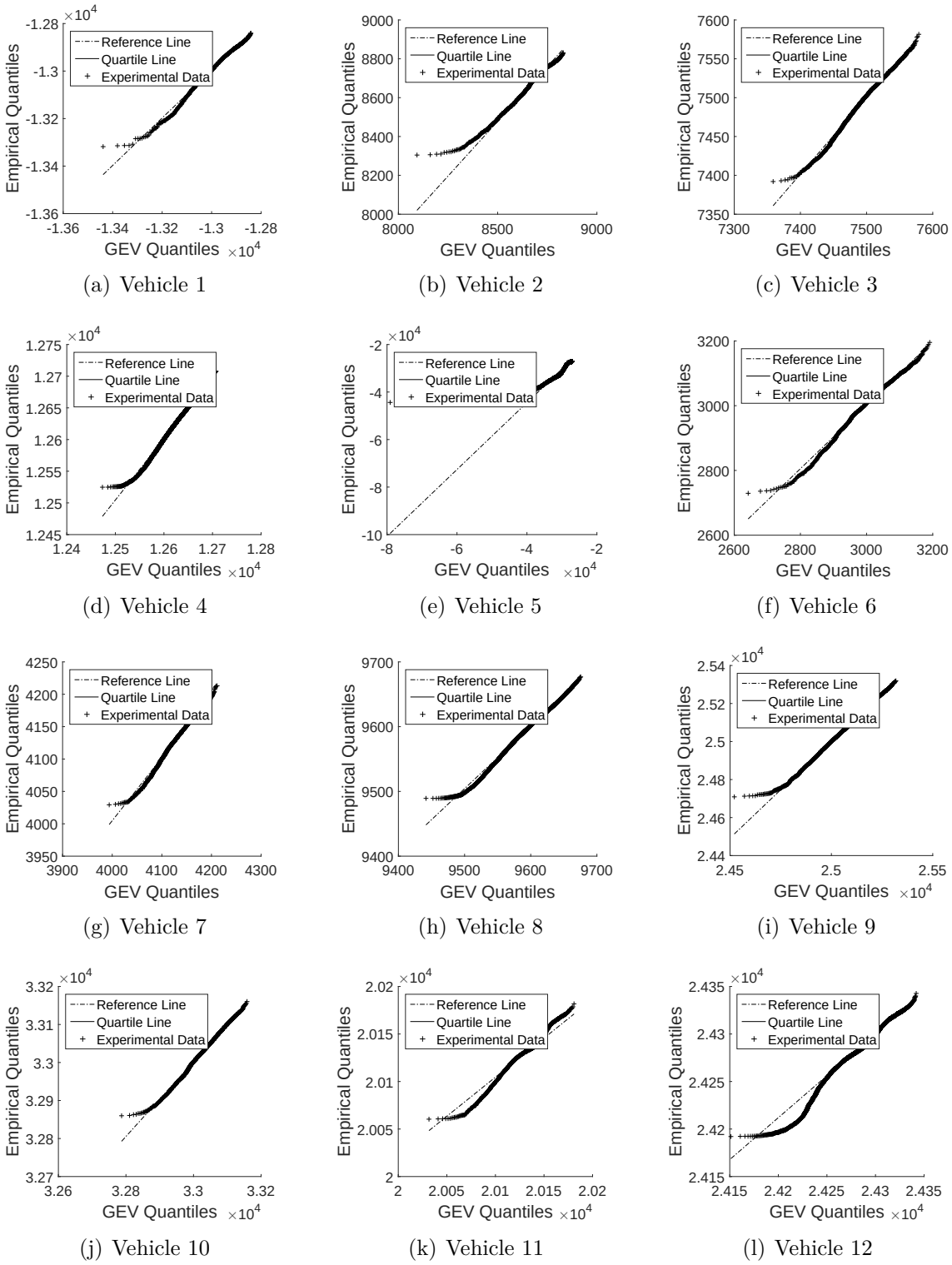
quantiles very well, even for Vehicle 5. This means the GEV distribution is not does not do a good job of capturing the distribution for lower values of the optimized results, but does do a good job at the upper values, which are the values of interest. A similar type of plot, a probability plot, shows the same result. These are included for each vehicle in Appendix D Section D.2.

The difference between the data and the hypothesized distributions can be quantified. Recall the discussion in the previous section regarding the KL divergence. Goodness of fit tests, such as the chi-square or the KS test, are designed to tell whether or not a set of data comes from a certain distribution, but not how far from a certain distribution the data is. A visual inspection of Figures 60 and 61 shows that the GEV distributions plotted fit the data much better than a uniform distribution or even a normal distribution would. However, the goodness of fit tests would give similar results if a uniform or normal distribution was applied. The KL divergence provides a way of quantifying how far a set of data is from a distribution. The KL divergence is calculated using Equation 84 [37].

$$d_{KL} = \sum_{i=1}^{d} P_i ln\left(\frac{P_i}{Q_i}\right) \tag{84}$$

Here, d represents the number of bins the data is divided into. In the continuous case, the sum can be converted to an integral. P_i and Q_i are the probability density values for the respective bins. A notional example is provided by calculating the KL divergence value for a set of data taken from a GEV distribution and compared to that same GEV distribution. This is repeated 1000 times to result in a set of KL divergence values, plotted in Figure 64(a). Instead of comparing to the GEV distribution, a normal distribution can be fit to the data and the KL divergence values calculated. These are shown in Figure 64(b). It can be seen that the KL divergence values are an order of magnitude larger when the data, which comes from a GEV distribution, is compared to a normal distribution.

The KL divergence values for the optimized data and the GEV distributions shown



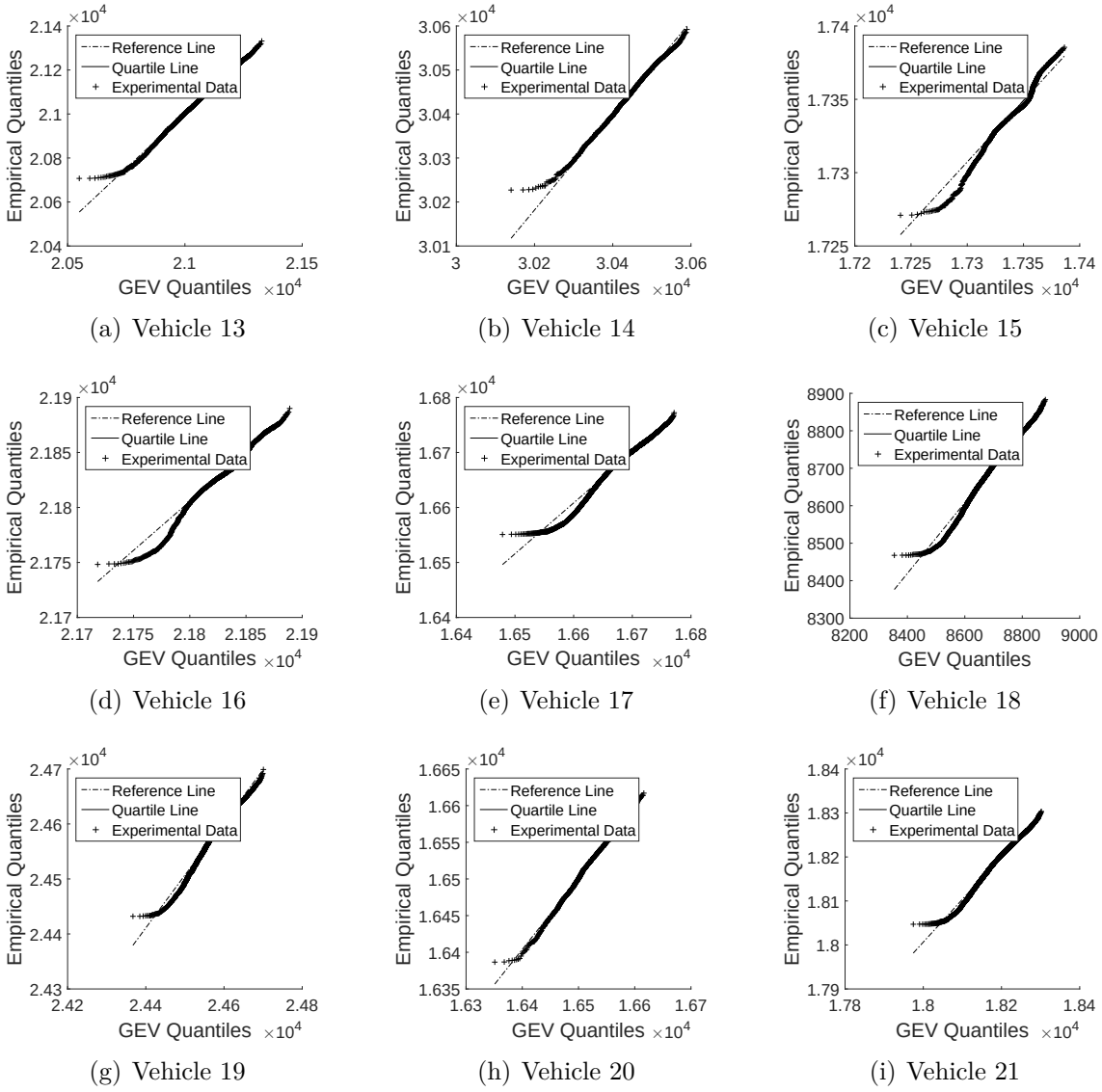
(a) Vehicle 1 (b) Vehicle 2 (c) Vehicle 3

(d) Vehicle 4 (e) Vehicle 5 (f) Vehicle 6

(g) Vehicle 7 (h) Vehicle 8 (i) Vehicle 9

(j) Vehicle 10 (k) Vehicle 11 (l) Vehicle 12

Figure 62: Q-Q plots for repeated optimization results for vehicles 1 - 12

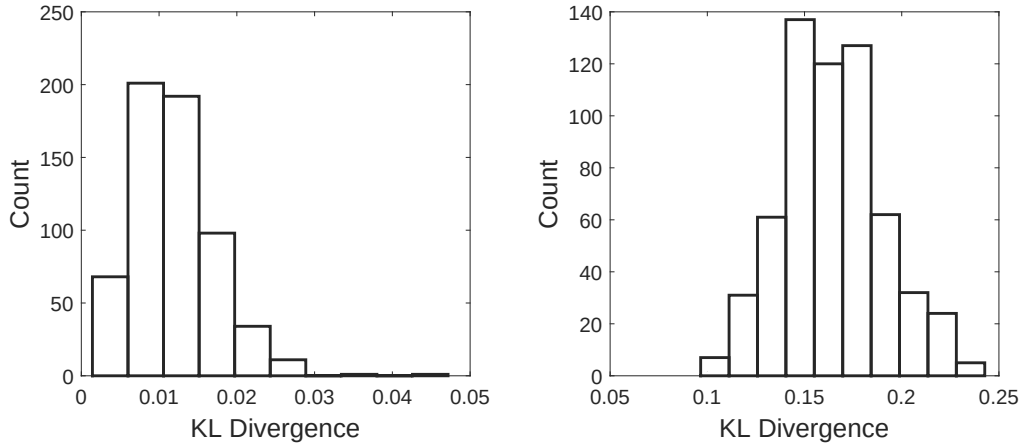


(a) Vehicle 13 (b) Vehicle 14 (c) Vehicle 15

(d) Vehicle 16 (e) Vehicle 17 (f) Vehicle 18

(g) Vehicle 19 (h) Vehicle 20 (i) Vehicle 21

Figure 63: Q-Q plots for repeated optimization results for vehicles 13 - 21



(a) GEV (b) Normal

Figure 64: KL divergence values for notional example

in Figures 60 and 61 are given in Table 16. When a GEV distribution is compared directly to itself, the KL divergence values are generally 0.04 or less, as seen in Figure 64(a). Given that the distributions considered here are already known to not be exactly GEV distributions, it seems reasonable to consider any KL divergence values of 0.08 or less as an acceptable approximation. Assuming that threshold, the values indicate that the GEV distribution is a good approximation of data for every vehicle except Vehicle 5. Again, this vehicle will end up being excluded but is considered for now. An argument may be made that an arbitrary threshold was chosen for the KL divergence. It should be noted, however, that this is being used as only one of several ways to determine if the GEV distribution is an appropriate approximation for the data.

Even though the goodness of fit tests do not suggest the GEV distribution provide good fits for the data, a visual inspection of the weight PDF's in Figures 60 and 61 and the Q-Q plots in Figures 62 and 63 indicate the GEV does approximate the data well, especially in the upper regions of the distribution. In addition, the KL divergence values indicate that there is little dissimilarity between the GEV distributions and the data. A final method of quantifying the GEV distributions in the data is provided

Table 16: KL divergence values for repeated optimization analysis and GEV fits

Vehicle	KL Divergence
1	0.020
2	0.027
3	0.016
4	0.006
5	0.159
6	0.020
7	0.015
8	0.009
9	0.002
10	0.006
11	0.063
12	0.061
13	0.005
14	0.012
15	0.079
16	0.055
17	0.037
18	0.015
19	0.043
20	0.007
21	0.020

by considering the value of a specific percentile.

The GEV distribution provides a way of approximating the distribution of performances resulting from repeated optimization analyses. The performance metric, however, should be a value, and not a distribution. The GEV distribution can be used to calculate a metric for the performance value. A simple process may be to use the absolute maximum, or 100th percentile. This may lead to inconsistent answers, however. Some GEV distributions go to ∞, which would lead to numerical issues, not to mention absurd results. The mean of the distribution could be used, but this would result in a significant offset from the actual performance capability of the vehicle. A middle ground can be found. In previous work, the author compared different percentiles for generating metrics from a GEV distribution in the context of trajectory optimization [161]. This study concluded that the 95th percentile provided a balance between approximating the maximum performance and yielding accurate results. For this thesis, the 95th percentile will be used as the metric from the GEV distribution. The error with respect to this metric between the optimized data and the GEV distribution can be calculated to quantify error. Table 17 shows the results. It can be seen that for all but Vehicle 5, the difference is less than 10 lb, and for most of the vehicles, the difference is less than 5 lb.

3.4.8 Answer to Research Question 4.2

The answer to Research Question 4.2 is that the distribution of results from repeated direct optimization analyses for a vehicle is not perfectly analogous the distribution of sample maxima. However, these two distributions are very similar. This was shown through a visual inspection of weighted PDF and Q-Q plots, the KL divergence values, and the difference in the 95th percentile for the set of representative vehicles. So while these two distributions are not the same, they are similar, especially in the upper regions of the distribution. Therefore, the GEV distribution is accepted as

Table 17: Difference in 95th percentile values (lb) for repeated optimization analyses and GEV fits

Vehicle	Difference in 95th percentile
1	6.8
2	8.1
3	2.8
4	1.4
5	35.1
6	9.3
7	3.9
8	2.4
9	0.6
10	2.2
11	0.7
12	2.1
13	4.4
14	4.9
15	1.5
16	0.7
17	5.7
18	7.6
19	9.4
20	2.6
21	6.1

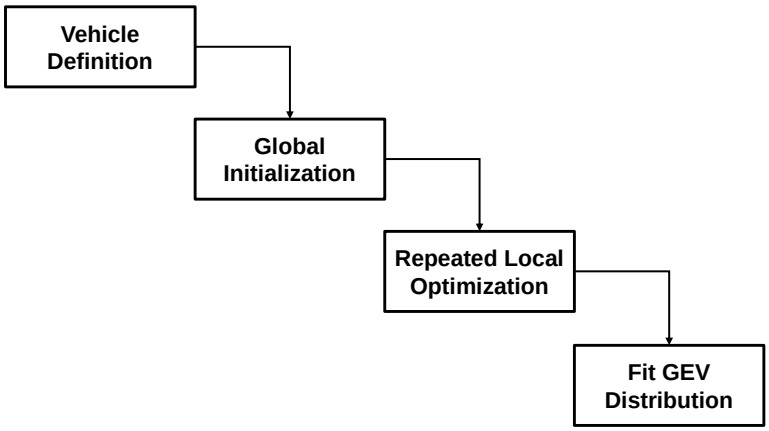


Figure 65: Method Buildup: Fit GEV distribution

an approximation of the distribution of repeated direct optimization analyses. This result will be employed in the final experiment to generate data across the entire design space of interest with the goal of creating a surrogate model.

At this point the next step in the methodology being developed can be added. After performing the global initialization and local direct optimization, a GEV distribution can be fit to the repeated optimization analyses. This process is shown in Figure 65.

3.5 Measuring Confidence

The statistical methods developed in the previous sections rely, as do most statistical methods, on a sample from the population. This means that there inherently exists error in the statistic that is estimated, which can lead to uncertainty. If there is a large enough sample, the error can be assumed to be small. As the sample size increases, the sample distribution approaches the population distribution. Knowing the sample size required, however, can be a difficult problem. Another common source of uncertainty is measurement error. In this thesis, it is assumed that the trajectory code used in this analysis will give accurate and precise data.

Figure 65 shows the method up to this point for finding the performance value for a vehicle. It involves using a global initialization of control vector guesses, direct

optimization for each of the points in the global search, and fitting a GEV distribution to the resulting data. In the previous section, this was considered analogous to the distribution of sample maxima. The GEV distribution is then used to find the performance value. Using repeated optimization simulations to generate the GEV distribution will greatly cut down on the computational effort for each vehicle. However, given that an inherently stochastic statistical process is being used, and that it is an automated method, a way of determining the confidence in the results is required. If the process is repeated it will result in a different set of values from the local optimization, analogous to sample maxima from a different set of samples. If the data is different the GEV distribution will be different, and the resulting reported performance different. So which is the actual best performance?

The answer is that most likely neither. Indeed the process can be repeated many times and many different "best" performances will result. (There is actually only one best performance, but finding that best performance with certainty is not currently possible.) The range of possible values for the answers is known as a sampling distribution [151]. This sampling distribution can be used to measure the confidence in the reported value using confidence intervals.

A confidence interval essentially states that the parameter being estimated lies between a pair of statistics with a certain probability. Equation 85 gives the mathematical definition, where L and U are the lower and upper bounds on the interval respectively, and $1 - \alpha$ is the probability of interest. A 95% confidence interval, for example, would have $\alpha = 10$. The number $1 - \alpha$ is called the confidence coefficient.

$$P(L \leq \theta \leq U) = 1 - \alpha \tag{85}$$

If the confidence interval is large, it means there is little confidence in the value of the statistic, whereas if the confidence interval is small, there is higher confidence. Ideally, the confidence interval would be 0, meaning the exact answer was known. In

the context of generating a surrogate model for launch vehicle performance, quantifying the confidence interval would be useful in determining if the data should be used or not. Data with large confidence intervals can be discarded or re-evaluated to increase confidence.

Finding the confidence interval is not always straight forward. The confidence interval applies to the statistics being estimated from the GEV distribution. Recall from Section 3.2.3 that there are several methods used to estimate statistics, including the maximum likelihood method, the probability-weighted moments, and the moment estimator [16]. For some special statistics, the confidence interval can be immediately calculated when certain methods are applied, but the results can be inaccurate [16]. These methods are not helpful in this context, so another method that relies on the sample itself and can be applied to any statistic is introduced.

3.5.1 The Bootstrap Method

The old saying "pulling yourself up by your bootstraps" is used to describe impossible tasks, where even extreme efforts will not yield results. Indeed the bootstrap method seems this way at first glance, but in this case it actually works. The method was introduced in 1979 by Efron [55]. Since then it has been widely developed and applied.

The business of statistics is to estimate properties of unknown distributions. This is done by using samples. For example, the mean of sample from a population is used to estimate the mean of the actual population. If another sample is taken, the mean will be different, even though it is still estimating the same population mean. Indeed if many samples are taken, what will result is a distribution of estimated means, known as a sampling distribution [151]. This sampling distribution can then be used to estimate the confidence in a certain value used to estimate the mean of the actual population.

The bootstrap method provides the same result using a single sample by re-sampling the sample. This means new samples is generated using only the data in the original sample. These new samples are referred to as surrogate data sets or surrogate samples (not to be confused with surrogate models). Because the data already exists, the surrogate samples are easily computed. These surrogate samples are obtained by randomly selecting values from the original sample with replacement (i.e. allowing values to be selected more than once). Each of these surrogate samples is then used to estimate the statistic of interest. What results, like before, is a distribution of estimates. This distribution can be used to obtain confidence intervals for the estimator from the original sample [151].

A key difference between using surrogate samples and actual samples is that actual samples can be used to get a better estimator and surrogate samples cannot. However, the goal here is to measure the confidence in the estimate, and bootstrap samples work perfectly for that. In fact, there are several ways to use bootstrapping to estimate the confidence intervals [146].

One of the ways bootstrapping is so beneficial is that it is computationally inexpensive and only requires already existing data. Given that trajectory optimization simulations are not cheap, this is very beneficial. Bootstrapping methods have been used before with EVT statistics [62, 95, 130] and are a perfect fit for application in this problem. The method is simple and fast, which makes it available for use for the thousands of vehicles to be evaluated. The application of the bootstrap method leads

to Research Question 5.

Research Question 5 - Does bootstrapping provide an accurate measure of confidence in the performance metric?

The idea here is to use the bootstrap method to determine if more data is required. After a set of optimization repetitions for a single vehicle has been obtained, the GEV distribution will be used to estimate the maximum performance of that vehicle. Bootstrapping can be used to generate a distribution of maximum performances from the bootstrap samples. Given that the bootstrap method has been used in conjunction with EVT before [62, 95, 130], it is expected that the bootstrap method will provide an accurate measure of the uncertainty in the performance metric of interest, as stated below in Hypothesis 5. It should be noted that the bootstrap method may result in false negatives, meaning an accurate performance measure results, but the uncertainty is high. In situations like this, that performance metric would be excluded or more repetitions could be run to decrease the uncertainty. The real test for the bootstrap method is in the number of false positives, i.e. performance metrics that are inaccurate but the uncertainty is low. For this reason, Hypothesis 5, stated below, is phrased to consider false positives.

Hypothesis 5 - If bootstrapping is used to measure uncertainty in the performance metric, then larger error in the performance metric will be indicated by higher uncertainty.

3.5.2 Experimental Setup for Research Question 5

The sixth experiment of this thesis will be conducted using the repeated optimization data generated in Section 3.4 for the set of representative vehicles from Section 2.6.3. A significantly higher number of optimization repetitions has been run for these test vehicles than would be required for a typical vehicle in this methodology. The performance metric derived from all the data available for these vehicles will be used to represent the global optimum. This performance will be denoted by y^*. In reality, it may not (and is likely not) the maximum performance, but because no analytical solution exists, it will be used as the maximum performance. A random subset, denoted A, of the repetitions can be used to represent what would be simulated for a typical vehicle being analyzed in this methodology. The GEV distribution can then be fit and the performance metric calculated. This metric will likely differ from y^*.

At this point, the bootstrap method can be applied to subset A (recall A represents the information available for a typical vehicle being evaluated). This involves repeatedly sampling with replacement from the subset A to generate bootstrap samples and estimating performance metric, in this case the 95th percentile. This will result in a distribution of performance metrics based entirely on the subset A. This distribution can be used estimate the uncertainty in the performance metric. This uncertainty will be quantified in two ways. The first will be the standard deviation. High standard deviation means that the bootstrap samples were very different from each other, and therefore there is high uncertainty. The second way to quantify uncertainty will be to measure the error between the performance metric from subset A, the original data, and the mean of the performance metrics from the bootstrap samples. Large error will again indicate uncertainty in the performance metric value.

It is important to recall from Section 3.5 that bootstrapping does not provide a way to find a better estimate for the performance metric if there is uncertainty. It is limited to quantifying uncertainty. If the uncertainty too large to be accepted, the

only way to proceed is to generate more data to reduce uncertainty or to exclude the data.

The subset A was taken from the repeated optimization simulations. The same process can be performed on a different subset from the same set of data, say subset B. Indeed this can be done repeatedly, and with different size subsets, for each vehicle. For this experiment, four different samples sizes will be considered: 25, 50, 100, and 200. For each of these samples sizes, 100 simulated samples will be taken from the repeated optimization data for each vehicle in the set of representative vehicles. The performance metric will be estimated using each of the samples, and the uncertainty will be quantified using the bootstrap method with 40 bootstrap samples. The performance metric for each sample will be compared to y^* to test Hypothesis 5.

The beauty of using bootstrapping for measuring confidence is that no further trajectory analysis is required. The data as it stands is used to estimate confidence. When the overall method in this thesis is implemented, if the confidence is below a certain level for a vehicle, more optimization repetitions can be used to find a better answer and/or increase confidence. What results is a feedback loop that allows for the method to increase the trajectory analysis in areas of the design space where it is needed.

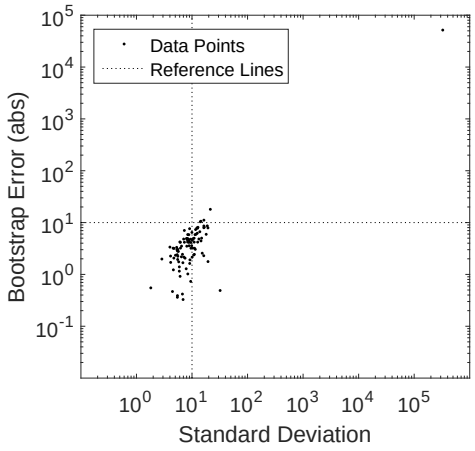
3.5.3 Experiment Results

The results from this experiment will be shown using two different types of plots. The first will show the standard deviation vs the bootstrap error. This error is the difference between the performance metric calculated using the sample and the average of the performance metric values that result from the bootstrap samples. These plots are shown in Figure 67 for Vehicle 1 for each of the different sample sizes. The reference lines are included to mark the value of 10 on each axis. Each

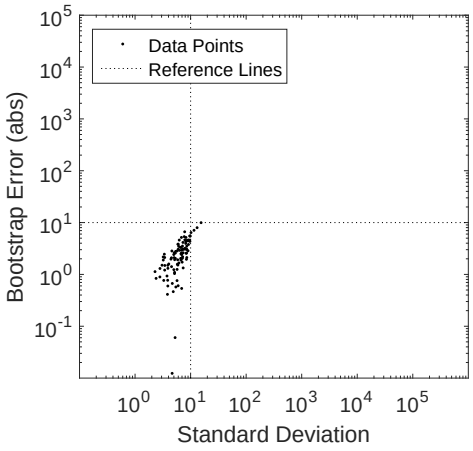
of the plotted points represents a subset of data that can be used to estimate the performance metric. After the bootstrap method is applied, the bootstrap error and standard deviation are plotted. In Figure 66(a) there is a single point in the upper right portion of the plot that represents a subset of data that did not estimate the performance metric with certainty. This resulted in high standard deviation as well as high bootstrap error. As the sample size increases from 25 to 200 it can be seen that the data points move to the lower left region of the plot with lower error and standard deviation, indicating higher certainty in the estimated performance metric. This is not unexpected, as more data will correlate to higher certainty. The other side of the coin is that higher sample size is more expensive.

The plots in Figure 66 can be used to essentially confirm that the bootstrap method can be applied here. In Figure 67 the bootstrap error is plotted against the error between the performance metric estimated by the sample and y^*, which will be referred to as the sample error. Recall y^* is the performance metric calculated using all the data available from Section 3.4.7. The sample error measures how much error there is between the performance metric when all the data is used and when a smaller sample size is used. Reference lines are included at a value of 10 for both axes. Low values of bootstrap error correlate to low values of sample error. The single point in Figure 67(a) in the upper right part of the plot represents the sample, discussed previously, that resulted in high uncertainty. As the sample size increases, the data tends to move farther down and to the left, representing lower error. Again, this is an expected result as more repetitions lead to lower error.

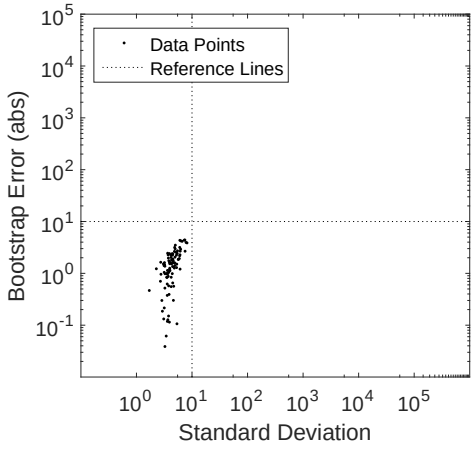
When the methodology implemented in this thesis is employed, the sample error will not be available, only the bootstrap error or the standard deviation. The bootstrap error can be used to indicate what value of sample error is typical. For example, in Figure 67(d), the bootstrap error values are all less than 10 and the sample errors are all less than about 20. Similar analysis of the plots for the other vehicles can lead



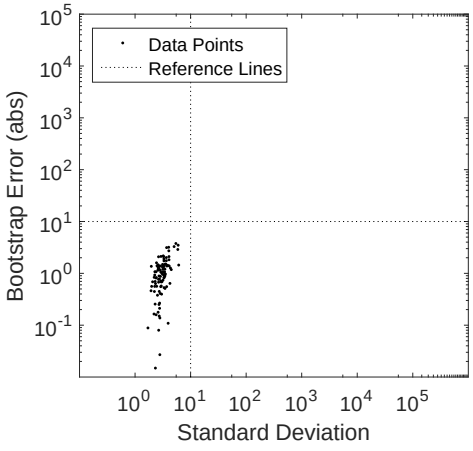
(a) 25 Data Points



(b) 50 Data Points

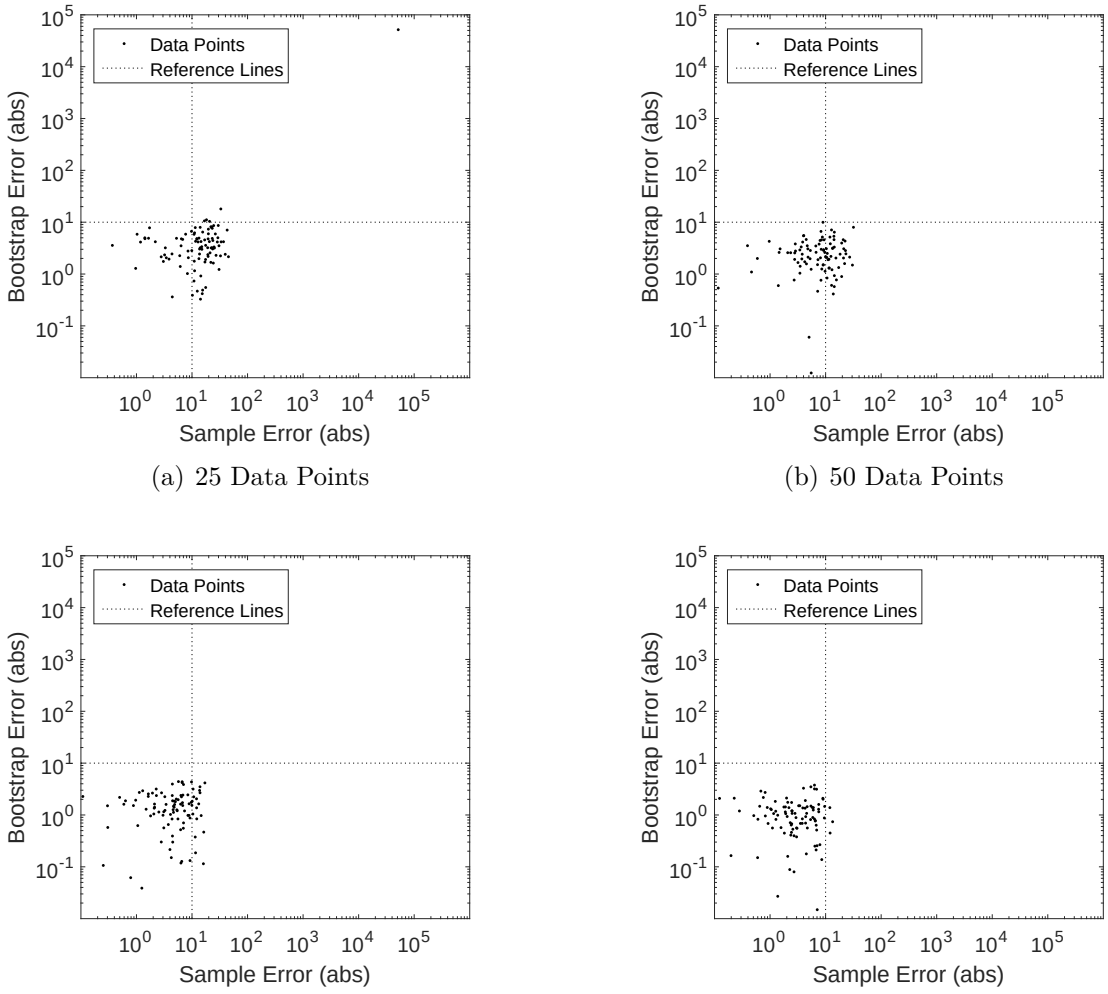


(c) 100 Data Points



(d) 200 Data Points

Figure 66: Bootstrap error vs standard deviation for Vehicle 1 using different sample sizes



(a) 25 Data Points

(b) 50 Data Points

(c) 100 Data Points

(d) 200 Data Points

Figure 67: Bootstrap error vs sample error for Vehicle 1 using different sample sizes

to a general rule for what value of bootstrap error should be accepted that results in acceptable sample errors.

Figures 68 and 69 show the bootstrap error vs standard deviation for all the vehicle when a sample size of 25 is used. Vehicle 5 in Figure 68(e) is an example where there was high standard deviation and bootstrap error for every sample. This means there is high uncertainty in any metric that results from this data. This vehicle would be discarded or more optimization repetitions analyzed before including the results. Vehicle 10 in Figure 68(j) is an example where some of the samples resulted in high

standard deviation and bootstrap error, while others did not. In every case, however, high standard deviation correlated to high bootstrap error.

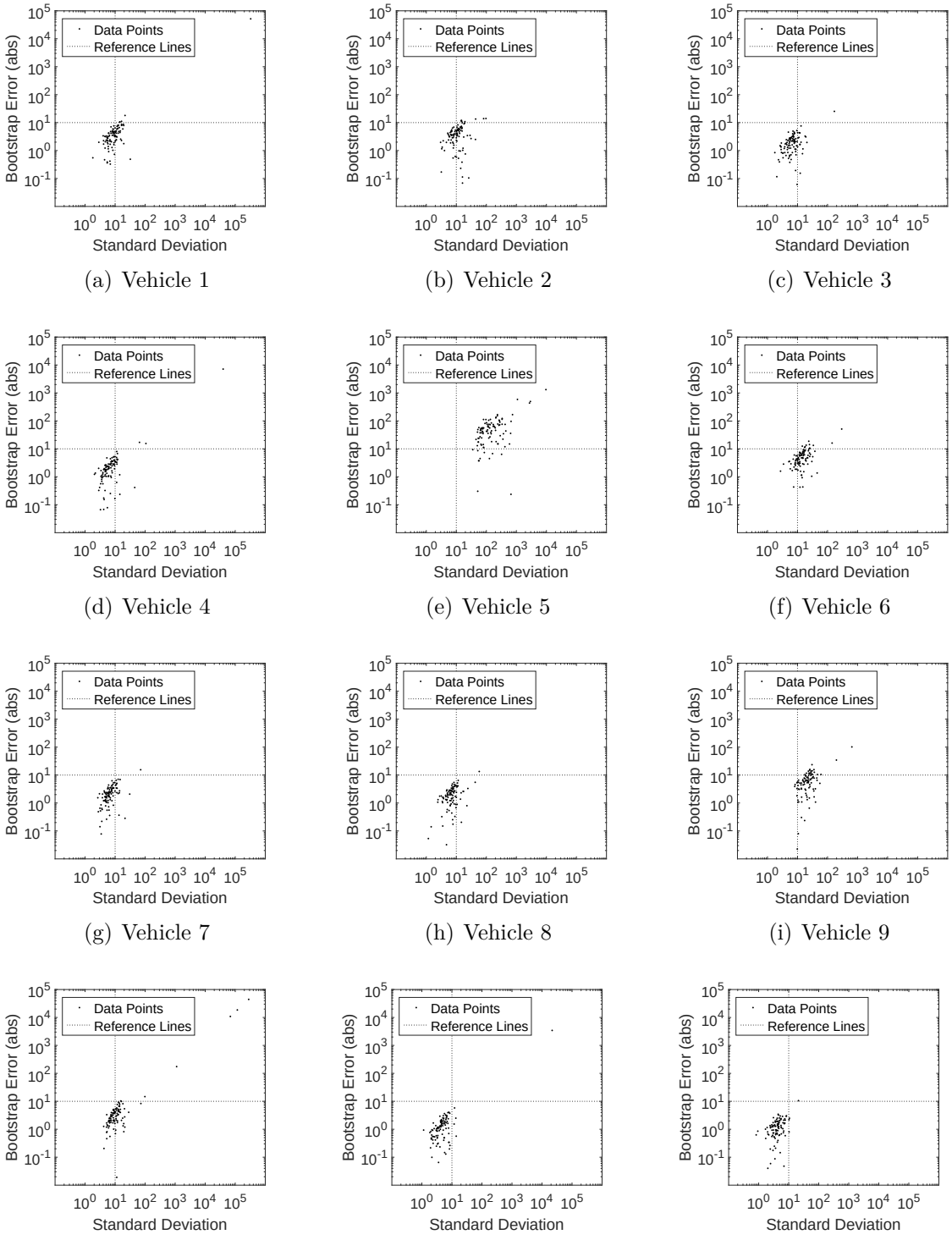
The bootstrap error vs sample error plots for each vehicle are included in Figures 70 and 71. Vehicle 10 in Figure 70(j) is an example of the correlation between sample error and bootstrap error. Recall that in the methodology, the sample error is not available. Bootstrap error can serve as an indicator of the sample error. For bootstrap error to be an indicator, the area in the lower right of the plots in Figures 70 and 71 should contain no data points. This area represents a situation where the bootstrap error is low, but the sample error is high. The goal is to screen out the data that leads to high sample error. The data for each of the set of representative vehicles indicates that bootstrap error is a good indicator, as there are no points in that region of the plots.

So far, the plots using a sample size of 25 have been shown for each vehicle. The plots when sample sizes of 50, 100, and 200 are used are shown in Appendix D Section D.3. As expected the data shifts to the lower left region of the plots as the sample size increases.

3.5.4 Answer to Research Question 5

The bootstrap method provides an accurate measure of confidence in the performance metric for each of the vehicles in the set of representative vehicles. This was stated in Hypothesis 5 by predicting the correlation of larger error with high uncertainty. The answer to Research Question 5 is that the bootstrap method does given an accurate representation of confidence, and therefore it will be used in the overall method to quantify the confidence. It should be noted that there exists the possibility of false negatives, or data sets that yield results with low error but high uncertainty. For this thesis, these situations will require more analysis, or the data will be excluded.

Figure 72 shows the method being developed in this thesis with all the elements



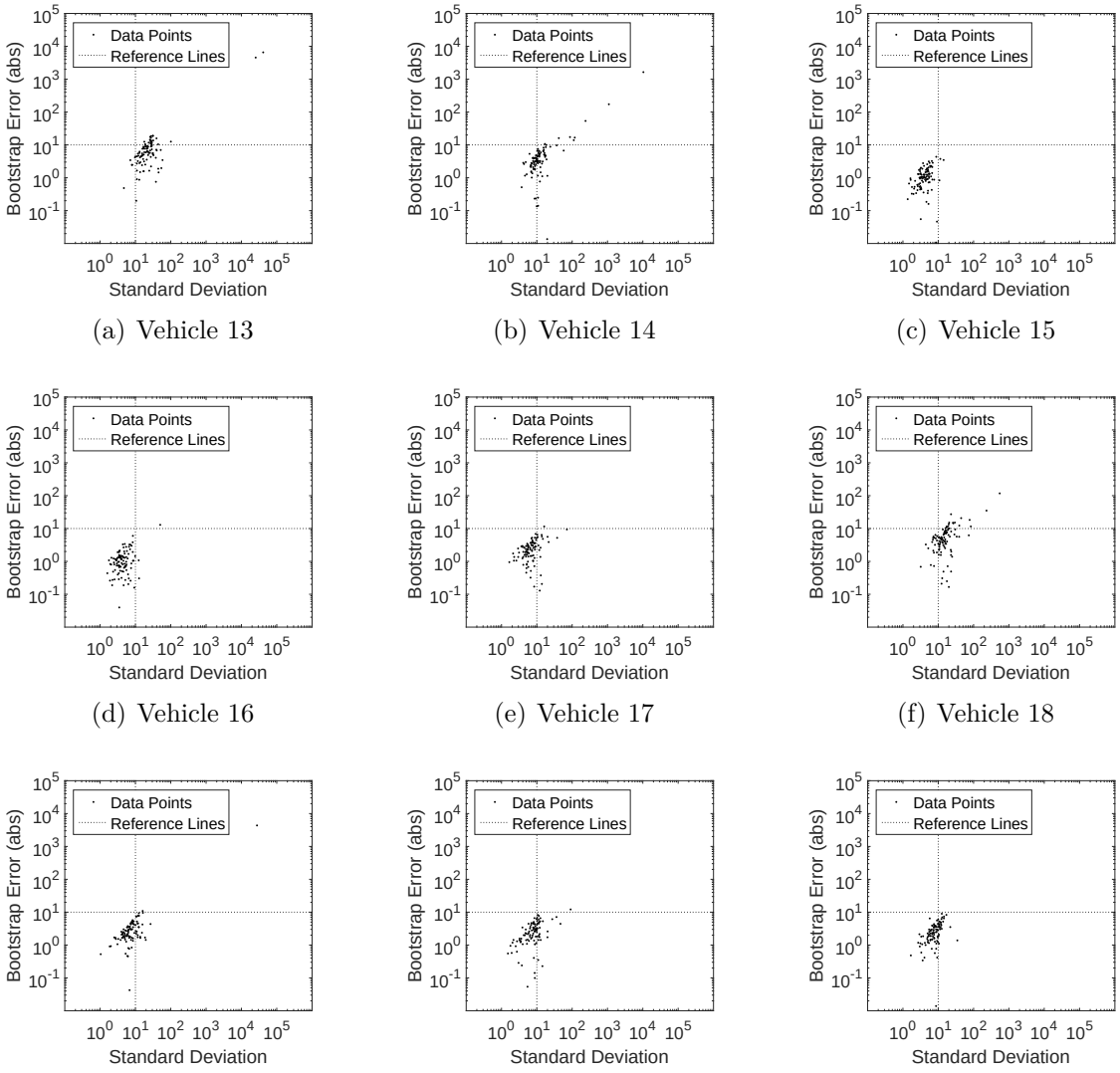
(a) Vehicle 1 (b) Vehicle 2 (c) Vehicle 3

(d) Vehicle 4 (e) Vehicle 5 (f) Vehicle 6

(g) Vehicle 7 (h) Vehicle 8 (i) Vehicle 9

(j) Vehicle 10 (k) Vehicle 11 (l) Vehicle 12

Figure 68: Bootstrap error vs standard deviation for vehicles 1-12 using a sample size of 25

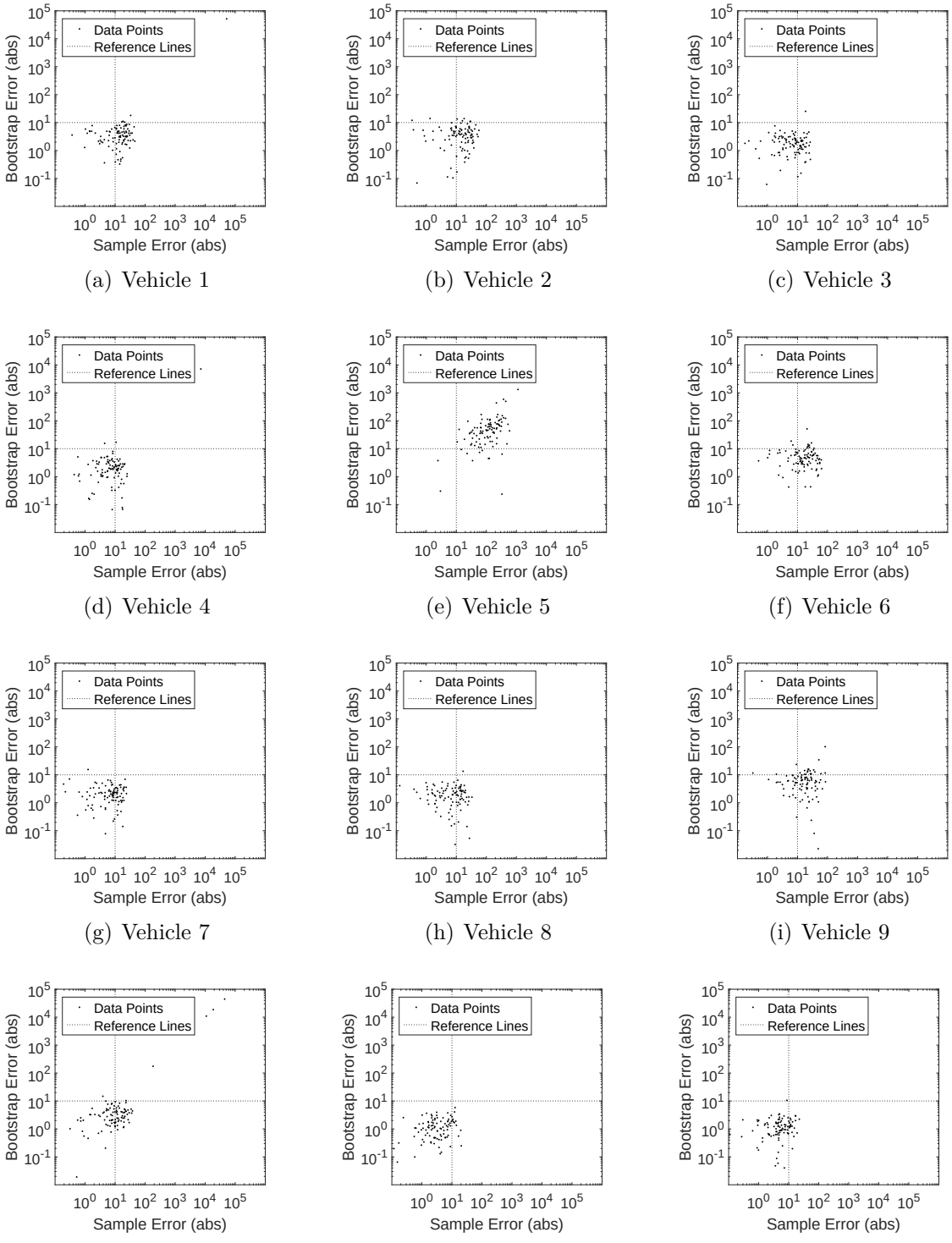


(a) Vehicle 13 (b) Vehicle 14 (c) Vehicle 15

(d) Vehicle 16 (e) Vehicle 17 (f) Vehicle 18

(g) Vehicle 19 (h) Vehicle 20 (i) Vehicle 21

Figure 69: Bootstrap error vs standard deviation for vehicles 13-21 using a sample size of 25



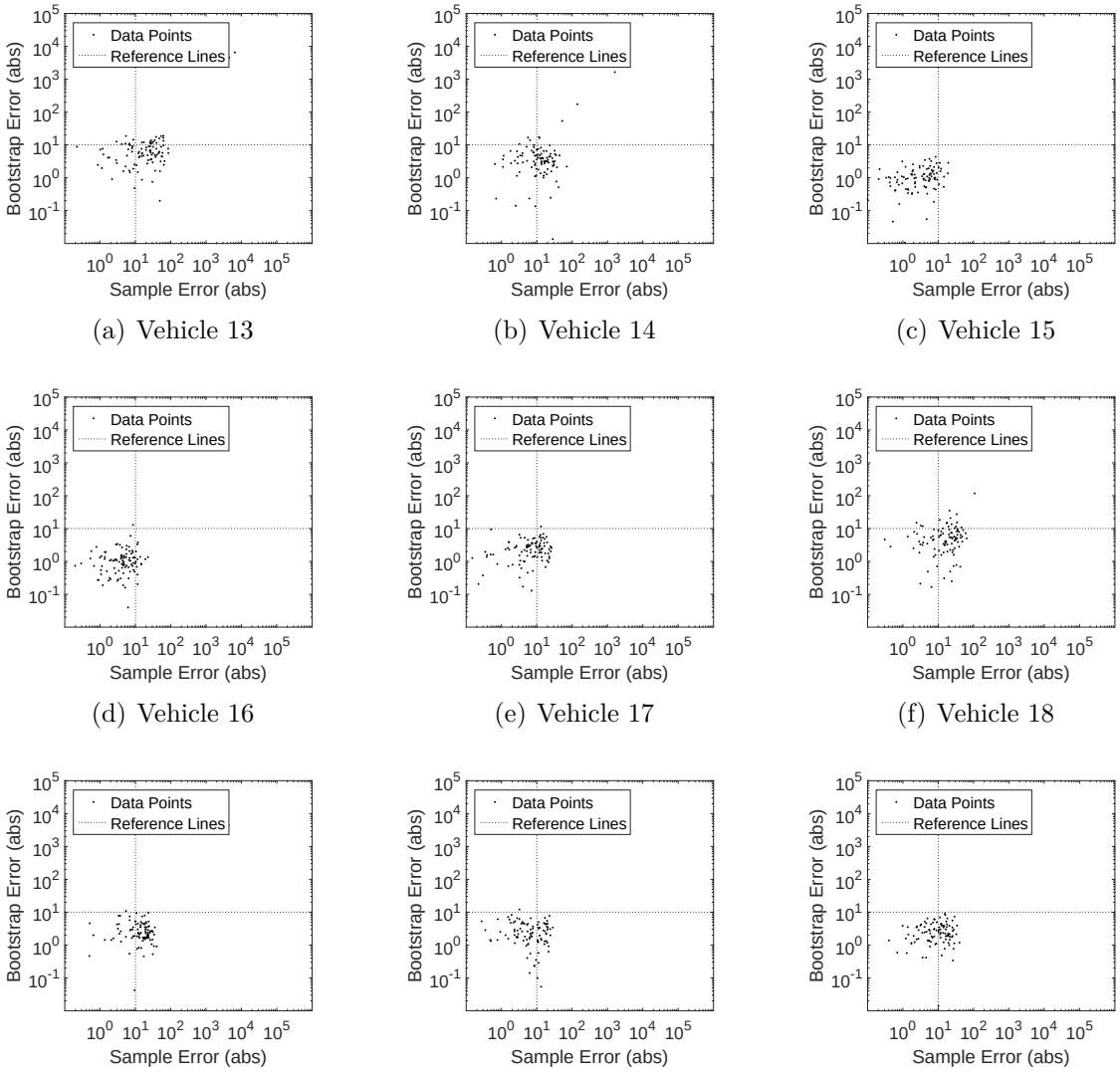
(a) Vehicle 1 (b) Vehicle 2 (c) Vehicle 3

(d) Vehicle 4 (e) Vehicle 5 (f) Vehicle 6

(g) Vehicle 7 (h) Vehicle 8 (i) Vehicle 9

(j) Vehicle 10 (k) Vehicle 11 (l) Vehicle 12

Figure 70: Bootstrap error vs sample error for vehicles 1-12 using a sample size of 25



(a) Vehicle 13 (b) Vehicle 14 (c) Vehicle 15

(d) Vehicle 16 (e) Vehicle 17 (f) Vehicle 18

(g) Vehicle 19 (h) Vehicle 20 (i) Vehicle 21

Figure 71: Bootstrap error vs sample error for vehicles 13-21 using a sample size of 25

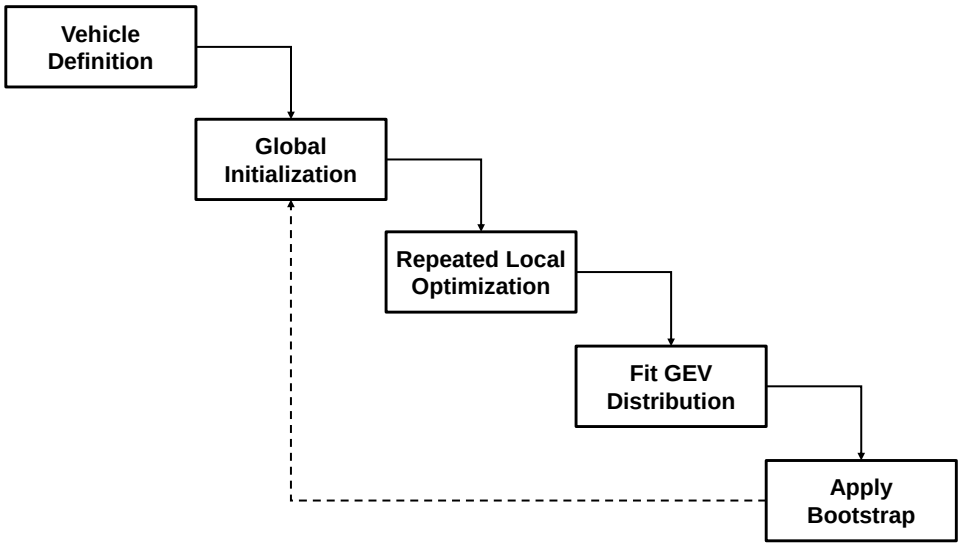


Figure 72: Method Buildup: Apply bootstrap method

discussed up to this point. The dashed line is included because the results of the bootstrap method may indicate the need for more data to be generated for a vehicle. In that case, another set of initial guesses, generated using a random search, would be optimized and a new GEV distribution fit. At this point, bootstrapping could be applied again with the additional data.

The process in Figure 72 represents a method that can estimate the performance of a launch vehicle. The elements of this method have been experimentally tested in Chapter 2 as well as this chapter. The goal of this thesis, however, is to enable design space exploration through the generation of a surrogate model. More than the performance of a single vehicle is needed for that. Up to this point this document has focused on how to find the performance for a single vehicle. Now the focus will shift to some of the elements required to generate a surrogate model.

3.6 Selection of Launch Vehicles Evaluation Set

Surrogate models were introduced in this thesis in Section 1.4. The generation of a surrogate model requires a set of data, in this case launch vehicle performance data. Once a method for evaluating the performance of each individual vehicle is

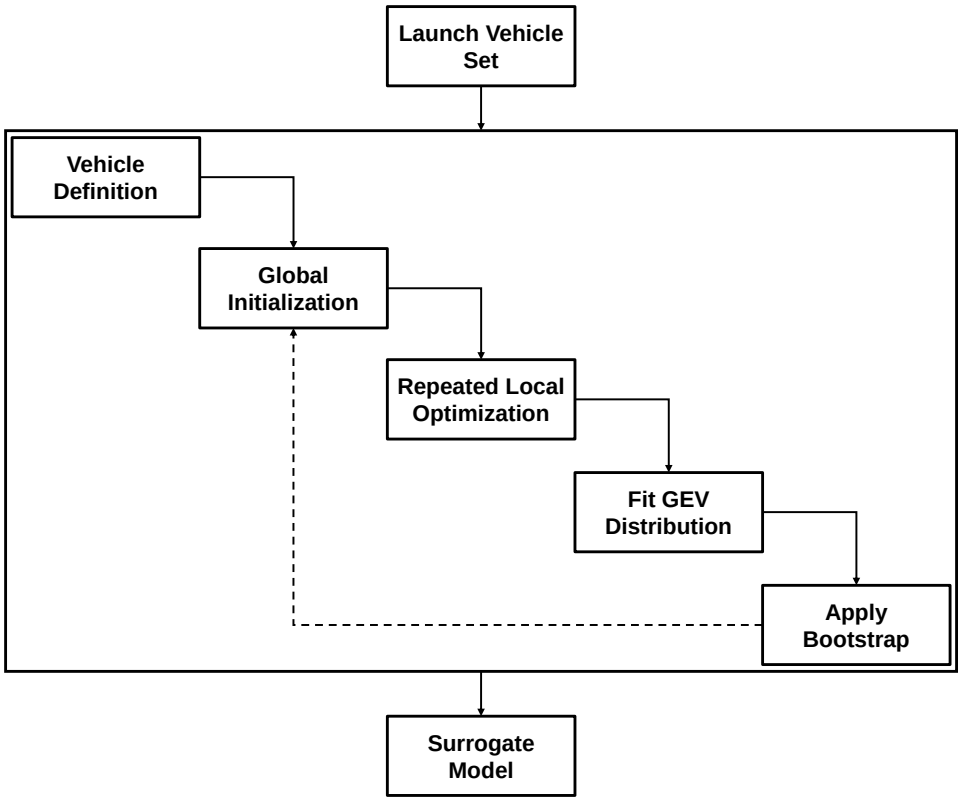


Figure 73: Method to develop launch vehicle performance analysis capability

determined, a set of vehicles to be evaluated must be selected to generate the data required. The data is then used to generate a surrogate model. This process is shown Figure 73. Once a surrogate model is created, it can be implemented in conceptual design to explore the design space, perform probabilistic assessments, evaluated technologies, and many other things. In addition, the surrogate model can be integrated into an MDAO environment.

The selection of the data to be evaluated leads to the sixth research question. This data refers to the different vehicles being considered in a conceptual design environment. The optimization of the trajectory is dealt with by the process in the

large box in Figure 73.

Research Question 6 - How should the alternatives to be evaluated be selected?

The field of statistics called Design of Experiments (DOE) is well suited for this problem. One of the goals of using statistics to design experiments is to maximize the amount of information obtained from a given set of experiments. In addition, data analysis is often simplified when using statistically designed experiments. Both of these characteristics are desirable in this problem. A more complete discussion on the subject is provided by Myers and Montgomery [136].

There are several ways to design experiments and a few of the options are presented here. This discussion is based on the lecture notes from the Advanced Design Methods class taught by Dr. Mavris at Georgia Institute of Technology in the Fall of 2010 [110] as well as the Engineering Statistics Handbook available online by the National Institute of Standards and Technology [120]. In general, there are two types of experiments. One type is a structured type, where the design will look like a structured pattern. The other type is random, where points are chosen randomly, but with some overall goal guiding the process. The following discussion will clarify this distinction.

Possibly the most intuitive design is known as a full-factorial design. This design is a structured design and involves discretizing the continuous variables and running every possible combination at each variable level. For example, if variables x_1 and x_2 are discretized into two levels each, a full-factorial design would involve 4 experiments: x_1 high with x_2 high and low and x_1 low with x_2 high and low. The number of experiments in a full-factorial is given by Equation 86.

$$\#\ of\ Experiments = \#\ of\ Levels^{\#\ of\ Variables} \tag{86}$$

This design does cover the design space, but with a lot of variables the number of experiments required can become infeasible. Suppose a problem has 5 variables. Two levels may not be enough to adequately map the design space. If 20 levels are used, the full-factorial design would require 3.2 million experiments. One hundred levels would require 10 billion experiments. If 20 levels are used with 6 variables, 64 million experiments are required. While the full-factorial design is good at covering the entire design space, it suffers from the "curse of dimensionality", where the number of experiments required becomes infeasible as the number of variables and levels increases. It is worth noting there exist fractional factorial designs, where not all the possible combinations are run. These, however, do not cover all the corners of the design space, and are not discussed any further in this document.

Central composite designs are another type of structured design that combine a 2-level full-factorial with a series of points at the midpoint of each variable range as well as a series of points with each variable but one at the midpoint of each variable range and the excluded variable set to its high or low value. Central composite designs are good at covering the corners of the design space, but may leave large areas of the interior empty.

There is a set of experimental designs known as space-filling designs. These designs are set up randomly. The first of these is known as sphere packing. The experiments are designed so as to be as far away in the design space from any other experiment as possible. This leads to a good coverage of the interior of the design space, but a lack of experiments in the corners.

Uniform designs are another type of space-filling designs. The goal of a uniform design is to have equal separation between all the points. Again, there is good coverage in the interior of the design space, but there can exist regions of the design space that have poor coverage.

The last type of space-filling designs discussed here is the Latin hypercube (LHC).

This design splits each variable into a number of bins equal to the number of experiments required. Then it guarantees that each bin of each variable will have a point in it. This method covers the interior of the space very well, but may not have points in the corner of the design space.

It is worth mentioning the concept of orthogonality at this point. An orthogonal set of experiments is one where the variables are linearly independent. With linearly independent variables there is no linear correlation between the variables. This is important in the design of experiments to ensure that any dependence seen in the outputs comes from the behavior of the system and not from what set of inputs were run. Full-factorial and central composite designs are inherently orthogonal. Space-filling designs are not, but can be set up in such a way as to minimize the linear dependence between the variables.

Given the wealth of available literature on the subject of Design of Experiments, Research Question 6 will be answered without an experiment. The goal is to sample the space with as few points as possible but still pick up on the performance trends with respect to the vehicle design parameters. A study comparing LHC and full-factorial designs for trajectory analysis for a specific vehicle showed that LHC designs characterized the space faster [162]. Other studies using design of experiments in the context of launch vehicles have also employed LHC designs [2, 3, 129]. In addition, given that little information is known about the vehicle performance problem, LHC are considered a good choice in the literature [120, 150]. Therefore, LHC designs are chosen to select the set of vehicles to be evaluated.

3.6.1 Design of Experiments for Trajectory Problem

The overall method proposed in this thesis is shown in Figure 73. So far, each element of the method has been tested through experimentation. At this point the final step before generating a surrogate model will be implemented: evaluate the set of launch

Table 18: Nested LHC design case numbers

Level	Cases
1	25
2	50
3	100
4	200
5	400
6	800
7	1600

vehicles. Research Question 6, which dealt with how this launch vehicle set should be chosen, was answered from the literature. A LHC design will be used to select vehicles. The vehicle design ranges for this problem are given in Table 5 in Section 2.5.2. The number of vehicles that should be evaluated is undetermined. For this reason, nested LHC designs will be used.

Nested LHC designs are designs that are themselves LHC, but a subset of that design is also LHC [131]. For example, if a 1600 case LHC design is created, it can be created in such a way that the first 800 cases are also a LHC design. This strategy is implemented here with sequential nested LHC designs. A total of 1600 cases is run. The smallest LHC design in this set is 25 cases. Table 18 shows the different levels and the corresponding number of cases. In total, only 1600 vehicles are evaluated, but 7 different LHC designs result from that data.

A global initialization is created and used for each of the vehicles in the design of experiment. This global initialization represents initial guesses on the control parameters. For this thesis, the global search consists of 500 initial guesses and the same set of initial guesses is used for each of the 1600 vehicles, resulting in a total of $800,000$ individual trajectory optimization analyses. Once the data is generated, GEV distributions can be fit for each vehicle. For some vehicles, as few as 4 analyses resulted in feasible trajectories. For others, as many as 280 analyses concluded with feasible trajectories. On average, this set of data contained about 75

Table 19: Number of available cases for different values of required repetitions

Required Repetitions	Cases
25	1224
50	857
100	430
200	46

feasible trajectories for each vehicle. The GEV distributions are used to calculate the performance metric, the 95th percentile.

The bootstrap method is essential for this process to filter out any vehicles for which there is uncertainty in the performance metric. For this thesis, it is required that the bootstrap error be less than 10 lb and the standard deviation of the performance metrics be less than 20 lb. In addition, limits can be set of the sample sizes, or number of optimization repetitions. Once these requirements are enforced, the total number of vehicles that can be used decreases from the original 1600. Table 19 gives the number of cases that meet the requirements listed in the paragraph with different limits on the number of repetitions. As the number of repetitions increases, the number of available cases decreases.

The cases, or vehicles, for each of the limits on the number of repetitions represent data that can be used to generate a surrogate model, which will be done in the next section. Figure 74 shows the input variables for the cases available when the number of required repetitions is set to 25. If each of the 1600 cases in the original LHC design was available, the points would form a perfect square in each of the plots. This is not the case, however. For lower values of US_TW and G_TW, for example, there are no feasible points. This is because this are of the input space represents vehicles that have a harder time reaching orbit. Throughout the other experiments in this chapter, Vehicle 5 has led to results that are not consistent with the rest of the vehicles. The reason for that is seen in this plot. Vehicle 5 has the lowest value for G_TW, US_TW, and CB_ISP and the highest for US_IMF of all the vehicles. If

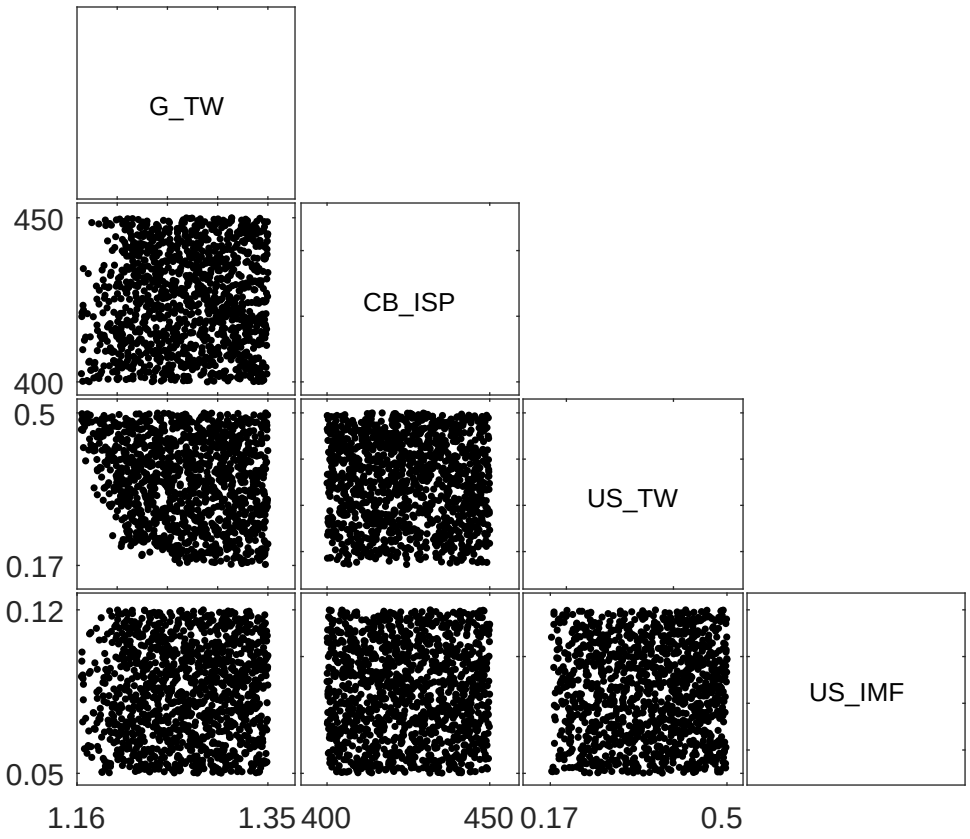


Figure 74: Scatter-plot of data available when the number of required repetitions is 25

these values are considered in Figure 74, many of those regions are characterized by a scarcity of data points. This lead to a difficulty in finding trajectories that reached orbit. Trajectories were found that reached orbit, however, all these trajectories had very large values for the control parameters. These values were so high that the trajectories become physically infeasible. Therefore, Vehicle 5 can be excluded, and none of the vehicles like it in the LHC design are included because there is no data for those vehicles.

3.7 Surrogate Modeling of the Performance Data

The key to achieving the research objective (from Section 1.3) is to find an analytic way to calculate launch vehicle performance. Surrogate models provide just that. Surrogate models, or metamodels, have been used frequently in engineering for a wide variety of applications, including launch vehicles [57, 85, 98, 150]. The term "metamodel", meaning a model of a model, is very apt. The goal of a metamodel is to provide an approximation for the analysis of interest for a very small computational cost. These approximations can then be used at the conceptual design level to explore the design space, a task that is not possible with more expensive engineering codes. Obviously, once a design is selected using a surrogate model, the engineering code should be used to obtain a more accurate result [150].

The next sections will provide a brief overview of some of the current surrogate modeling techniques. There are a wide variety of options and each have their associated advantages and disadvantages.

3.7.1 Surrogate Modeling Techniques

3.7.1.1 Polynomial Regression

Polynomial regressions, or response surfaces, were first developed in the 1950's by Box and Wilson [26]. Linear regression is used to find the coefficients of a polynomial that best fit the data. The typical second-order polynomial used is shown in Equation 87.

$$\hat{y} = \beta_0 + \sum_{j=1}^{k} \beta_j x_j + \sum_{j=1}^{k} \beta_{jj} x_j^2 + \sum_{i=1}^{j} \sum_{j=2}^{k} \beta_{ij} x_i x_j \tag{87}$$

In this type of regression, $\hat{y}$ is the response predicted by the surrogate model and the x terms are the input variables. For the launch vehicle problem $\hat{y}$ would be the performance and the $x's$ would be vehicle design parameters, like thrust or weight. The β terms are the coefficients of the surrogate model. These are used to fit the data. The process of fitting a surrogate model is essentially finding the $\beta's$ that provide the

best estimates, $\hat{y}$ for the given inputs, $x's$.

Polynomial regressions are generally associated with Response Surface Methodology, which involves a set of screening and fitting steps to find a good metamodel, and are commonly used in that context [117]. Any order polynomial can be used, depending on the complexity of the underlying problem and the amount of data available. As the polynomial order increases, more points would be necessary to fit the polynomial.

Polynomial regressions are probably the most common and easiest surrogate modeling method to implement [150]. The drawbacks of this method are that it is not suited to high-dimensions, and it does not perform very well for highly nonlinear problems [85].

3.7.1.2 Neural Networks

Neural networks are commonly used when very little is known about the form of the underlying problem. A neural network consists of a set of "neurons" organized into an architecture that send signals to each other to produce an output. Figure 75 shows a sample neuron architecture.

The inputs are feed into the system in the input layer. Each neuron then performs a calculation and passes its output on to the next set of neurons. The function in each neuron can be linear or nonlinear. The output layer is the resulting value of the neural net [93]. The overall set of neurons is what makes up the neural net architecture. In principle, any architecture is possible, and any number of neurons is possible. In practice, the more neurons, the more computationally intensive the fitting process is.

Neural networks can be slightly more difficult to understand than polynomial regressions, but the principle is the same. Each neuron will have some function with coefficients that define how the overall architecture behaves. The process of fitting a neural net is finding the appropriate coefficients. Neural net fitting is usually a

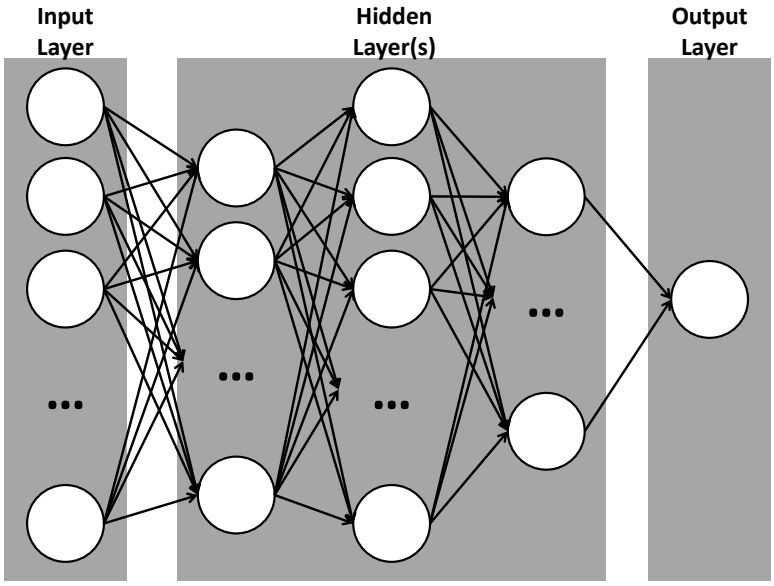


Figure 75: Example neural network architecture

stochastic process, and many different neural nets (i.e. different in architecture or coefficients) can yield good fits.

One of the challenges of using neural networks is setting the number of nodes and hidden layers, i.e. defining the architecture. These selections can have a large effect on the model performance [100]. In addition, neural networks can require a large amount of data in order approximate the problem well. The big benefit of neural networks is that they work well for highly nonlinear problems, and so with enough data can find good approximations [150].

3.7.1.3 Kriging

Kriging is a metamodeling method that models the output as some known function $f(x)$ plus a correction factor, as seen in Equation 88 [85].

$$\hat{y} = f(x) + Z(x) \tag{88}$$

Usually, $Z(x)$ is considered the realization of a stochastic (usually Gaussian) process [150]. The $f(x)$ term is a global model, while the $Z(x)$ is a local deviation from the global model that can be used to fit the data perfectly.

Developing a kriging model can be complex, but it works well for a large variety of problems [85, 150] and has been considered as an alternative to other surrogate modeling methods for aerospace applications [149].

3.7.1.4 Radial Basis Functions

A radial function is a function that only depends on the distance between the input argument and the defined origin. For example in a 2-D plane, set $f(r) = r^2$ where $r = \sqrt{x^2 + y^2}$. The function $f(r)$ only depends on the distance of the point (x, y) from the origin. Any point on a circle centered on the origin would yield the same function evaluation.

A radial basis function is a linear combination of radial functions h, as seen in Equation [100].

$$\hat{y} = \sum_{i=1}^{n} w_i h_i(x) \tag{89}$$

Here, w is a weighting parameter and n is the number of basis functions.

Radial basis functions were developed originally for interpolation of scattered multivariate data [85]. They can be thought of as a network of nodes, much like a neural network, but based on the function in Equation 89 [30]. A survey of these functions is given by Buhmann [33].

3.7.1.5 Multavariate Adaptive Regression Splines

Multivariate adaptive regression splines (MARS) use a linear combination of a set of basis functions to approximate the response, similar to radial basis functions except the basis functions do not have to be radial. MARS splits the design space into regions and fits a specific regression to each region. One advantage of MARS is that it makes no assumptions about the underlying problem, and therefore can be flexible [100]. The disadvantage is that it is relatively new and can be difficult to implement [85].

3.7.1.6 Support Vector Regression

Support Vector Machine (SVM) or Support Vector Regression (SVR) is a method that developed from machine learning theory. There are many different variations of SVR, but the basic idea is to find a function that approximates all the data within a certain error. As long as the approximation is within the prescribed error, the algorithm is satisfied. There is no attempt to minimize any error within the prescribed range. SVR functions can take the form of Equation 90.

$$f(x) = \langle w, x \rangle + b \tag{90}$$

where $\langle w, x \rangle$ represents the dot product between w and x. Other forms of SVR functions are introduced to allow some error or deal with nonlinearities [154]. SVR is very useful when the underlying function of the process used to generate the data is unknown [13]. SVR is relatively new in the field of surrogate modeling, but it has been applied to several different engineering analyses [42].

3.7.2 Ensuring a Good Surrogate Model Fit

Regardless of the type of surrogate modeling technique, the goal is to accurately predict what the underlying analysis code would output for the inputs of interest. Inherent in the creating of the surrogate model is a way of checking how accurate the surrogate model is, a goodness of fit check. This accuracy assessment can be checked in several ways. Corman and German give a good description of some of the relevant metrics [47].

The R^2 is a measure of how well the surrogate model captures the variance of the actual data. Ideally, the R^2 value, sometimes called coefficient of determination, is very close to 1. The adjusted coefficient of determination takes into account the number of data points used to fit the model. The formulas for both of these are shown in Equations 91 and 92 [47].

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2} \tag{91}$$

$$R^2_{adj} = 1 - (1 - R^2)\frac{n-1}{n-k-1} \tag{92}$$

where y is an observed value, $\hat{y}$ is the predicted value, $\bar{y}$ is the average of the data, n is the number of data points, and k is the number of design variables used in the model.

The normal and adjusted coefficients of determination do not guarantee a good fit even if they are close to 1. Model fit error (MFE) and model representation error (MRE) are two other metrics that should be considered. Model error is the difference between the observed value and the predicted value. The difference between fit and representation error is that MFE only considers the points used to fit the model. MRE considers points that were not used to fit the model. The goal of a surrogate model is to be able to predict an outcome for a case without ever having seen that case before. It is not unusual to exclude a certain percentage of the cases when fitting a model[76]. For example, the model can be fit using only 70% of the data. The other 30% is used to ensure the model has predictive capabilities. A model with low MFE but high MRE is not a good model. MFE and MRE are easily evaluated in graphical manner by plotting the residual on the y-axis and the predicted value on the x-axis.

Two other metrics are the average and maximum absolute error. The average absolute error (AAE) takes the average of all the residuals and divides by the standard deviation times the number of points. The maximum absolute error (MAE) divides the maximum residual by the standard deviation. AAE measures overall model performance while MAE gives a measure of the worst case scenario. Equations 93 and 94 show AAE and MAE respectively.

$$AAE = \frac{\sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{\sigma_y} \right|}{n} \tag{93}$$

$$MAE = max|\frac{y_i - \hat{y}_i}{\sigma_y}| \tag{94}$$

The final metric for goodness of fit discussed here is the root mean squared error (RMSE). RMSE is a measure of how far off the model is from the actual data. A good fit will minimize RMSE. The equation for RMSE is shown in Equation 95.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{\sigma_y}\right)^2} \tag{95}$$

These metrics can be used to measure the accuracy of a particular fit. There have been a few studies comparing how well a specific type of surrogate model works with general problems [85, 100, 172]. Other metrics for these types of studies can include robustness, efficiency, transparency, and flexibility. These metrics are important when developing surrogate models for general problems. For example, it is important that a surrogate modeling technique be flexible so it can be used in many different applications. For this thesis, however, the problem is specified, so the only metric of interest is the accuracy.

3.7.3 Surrogate Models in Launch Vehicle Design

There have been several applications of surrogate models in launch vehicle design to this date. In 2012 Lafleur et al [96] generated surrogate models for the payload capacity of a given launch vehicle to orbit. This data was collected from payload planners guides. The surrogate models, all polynomial regressions, mapped the maximum payload to the specified orbit altitude and inclination. The utility of this is obvious in the context of spacecraft design given an existing launch vehicle. This problem, however, is inherently different than the one addressed in this thesis because trajectory information was already available and simply gathered from the literature.

Another more complicated study was conducted in 1993 by Engelund et al [57]. The goal was to find the best aerodynamic configuration for a single stage to orbit vehicle. The design variables considered were all geometric, varying shapes of the

fuselage, nose and wing. The study used an integrated design tool with capability to analyze geometry, structures, aerodynamics, propulsion, trajectory, and heating. The trajectory analysis was performed using POST, which is the tool selected for the trajectory analysis in this thesis (see Section 2.4).

Using POST, or any current trajectory optimizer, in the loop was made possible by the small variations being considered for this vehicle. This study considered changes to the aerodynamic configuration. While aerodynamics have a large effect on the overall trajectory, changes in the variables considered did not have a large effect on the trajectory. So once a baseline was found, it worked for all the variable combinations being considered.

In 2005 Lee et al published an effort to create a surrogate model for the nose fairing of a launch vehicle [98]. The goal was to minimize drag while constraining heating and the volume inside the nose fairing. The generation of the surrogate model was included in the design process. An initial set of data was used to create a surrogate model. The optimum was found using the surrogate model, and then rerun using the actual analysis code. If the surrogate model and analysis code outputs converged, the solution was output. Otherwise, a new surrogate was created using the extra point and the process was repeated until convergence.

This method of including the surrogate modeling analysis in optimization process does not lend itself to most problems. First of all, it may be computationally inefficient to fit surrogate models every iteration. And secondly, regression issues may arise that lead to a particular problem never converging. This study does illustrate an important point, however. Regardless of how accurate a surrogate is, any vehicle selected for the next design phase using a surrogate model should always be evaluated using the underlying analysis tool to confirm that it is indeed a desirable design.

The most relevant example of surrogate modeling of launch vehicle trajectories is from work that Linshu was involved with [2, 3, 129]. While he is not first author on

any of the papers, he is involved in three very similar studies considering SVM as a regression option for trajectory and vehicle metrics in conceptual design. Two of the studies involve surrogate models of trajectories rather than performance. In both cases, the trajectory models are greatly simplified, using only 2-D simulations and spherical non-rotating earth models. In addition, none of the studies measured how well the optimization technique employed found globally optimum trajectory. Even so, these studies do show an interest in being able to find a way to quickly obtain launch vehicle performance data in conceptual design.

All of these examples are fundamentally different from the method proposed here. The closest example is the last one described, the work from Linshu [2, 3, 129]. However, this work is trajectory analysis. The goal was to map performance to vehicle *AND* trajectory variables. The work here is to exclude trajectory variables from the final model, mapping vehicles directly to performance. The motivation for this was presented in Chapter 1.

3.7.4 Surrogate Model Selection

The final step in the methodology in Figure 73 is to generate a surrogate model using the data. A surrogate model allows for the evaluation of hundreds of thousands of vehicles in the context of conceptual design. The motivation for this was introduced in Chapter 1. Different types of surrogate models can be used, several of which have been discussed in the preceding sections. Selecting the right surrogate model is an important part of this process. If the form of the surrogate model does not apply to the underlying problem, fitting the data will be impossible. Choosing the type of

surrogate model leads to Research Question 7.

Research Question 7 - Which surrogate modeling technique should be used for launch vehicle performance data?

The launch vehicle trajectory problem is expected to be nonlinear. This means that certain types of surrogate models, such as polynomial regressions, may not work very well. The problem does not display the inherent geometric characteristics suited to radial basis functions. Given the nature of the problem, and the robust nature of neural networks, it is expected that these types of models will provide the best surrogate model approximations, as stated below in Hypothesis 7.

Hypothesis 7 - If a neural network is used to create a surrogate model for the performance data, the neural network will provide a better fit to the data than polynomial regression, kriging models, radial basis functions, or support vector regressions.

This question will be addressed using an experiment in the following sections. This experiment will compare how well the surrogate models mentioned in the hypothesis fit the given data. Various measures for testing the goodness of fit of a surrogate model were given in Section 3.7.2. For this experiment, R^2 and RMSE will be used for both fit data and validation data.

Table 20: Number of fit and validation cases for nested LHC designs with different number of required repetitions

LHC Design	25 Repetitions		50 Repetitions		100 Repetitions	
	Fit	Validation	Fit	Validation	Fit	Validation
25	18	5	10	2	4	0
50	34	9	21	5	10	2
100	58	23	33	16	17	6
200	106	51	64	33	35	14
400	208	103	127	69	67	31
800	407	200	272	127	136	64
1600	845	379	594	263	290	140

3.7.5 Experimental Setup for Research Question 7

Five different types of surrogate models will be fit using the data from Section 3.6.1. The models will be fit using cases with the required number of repetitions set to 25, 50, and 100. When the required number of cases is set to 200, only 46 cases result. This is too low to fit a surrogate model with so it is excluded. In addition, the surrogate models will be fit using each of the nested LHC designs. Each of the seven designs is shown in Table 18. The number of cases available, however, will be smaller than these numbers, as some of the vehicles did not result in enough feasible trajectories. Combining the number of repetitions required with the nested LHC designs results in a total of 21 individual surrogate models for each of the five model types.

For each of the fits, validation data will be randomly selected from the data available. For this study, 30% of the cases will be set aside as validation data. This means the surrogate model will be fit with no knowledge of the validation cases. The validation cases are chosen and kept the same for all the surrogate fits. For example, if a case in the smallest LHC design is a validation case, it will also be a validation case for the other LHC designs. The validation cases are also kept the same for different model types. Table 20 shows the number of fit and validation cases for each of the nested LHC designs for the different values of required repetitions.

The fits will be evaluated using the RMSE and R^2 metrics, introduced in Section

3.7.2. In addition, the MFE and MRE will be shown using two types of plots. The first will plot the actual value against the predicted value. For a perfect fit, this would result in a straight line. This can be done separately for the data used to fit the model and the data used to validate the model. The second type of plot will show the residual error (actual value - predicted value) against the predicted value. This is useful in identifying any trends in the residual error.

3.7.6 Experiment Results

3.7.6.1 Neural Networks

The goodness of fit metrics evaluated for this experiment for each of the neural networks are shown in Tables 21 and 22. The "Fit" and "Validation" columns indicate the metric was calculated using the data used to fit or validate the model, respectively. The R^2 values indicate very good fits, especially for the larger LHC design. The RMSE values are also very good. Recall the values being predicted by the surrogate model are on the order of $10,000$ lb, so an RMSE value of 5.19, for example, is exceptional. The number of points used should also be taken into account when evaluating the surrogate models. When the LHC design with 25 cases is used and 25 repetitions are required, the RMSE and R^2 values are reasonable. Referring to Table 20, only 18 points were used to create the model and only 5 were used to validate. This is far too low to expect a reasonable approximation of the design space. Later on, this will be shown by including all the points available to validate the fit. The fit using the LHC design of 25 cases and requiring 100 repetitions shows a "NaN" in for the validation metrics. This is because none of the points in this DOE were chosen as validation points, and therefore no validation metrics are available. In practice, this is not a good way to generate a fit, but the data is included in the tables for completeness. Because the same validation cases are used for all the types of models, no validation metrics will be available for this set of data.

The RMSE values tend to be higher when fewer cases are used to create the model.

Table 21: RMSE values for the neural networks

LHC Design	25 Repetitions		50 Repetitions		100 Repetitions	
	Fit	Validation	Fit	Validation	Fit	Validation
25	1.63	367.89	1.09	2.57	0	NaN
50	9.4	102.18	1.23	43.74	0	15.92
100	10.27	48.1	19.77	53.31	18.96	58.53
200	10.56	36.95	10.71	15.98	6.03	21.39
400	6.85	20.68	3.38	8.27	1.14	16.3
800	7.55	12.53	5.06	8.25	2.83	7.99
1600	8.48	10.03	5.1	6.21	3.3	5.19

This is especially true when RMSE is calculated using the validation points (except when the number of validation cases is very small). This is an expected result. More cases generally leads to a better model. The trend is less clear as the number of required repetitions increases. There are two competing influences at work. When the number of required repetitions is kept low, there are more cases available to fit the model with. This is shown in Table 19 for all the data, and it is seen for each of the LHC designs in Table 20. However, when the number of required repetitions increases, each data point has less uncertainty, as was shown in Section 3.5. The data can be thought of as less "noisy". Consider the extreme example of only using a single repetition for each case. This is analogous to randomly choosing a performance from the GEV distribution that results from repeated optimization, as was explained in Section 3.1.1. The surrogate model must account for trends based on this random behavior as well as the trends that result from the physics of the problem. For this reason models sometimes get better as the repetitions increases (compare the LHC design with 800 cases and 25 required repetitions vs 50 required repetitions) and sometimes gets worse (compare the LHC design with 800 cases and 50 required repetitions vs 100 required repetitions). Similar trends will be seen with the other surrogate model types.

The actual by predicted and residual by predicted plots for the LHC designs with 1600 cases and 100 required repetitions are shown in Figure 76. The actual by

Table 22: R^2 values for the neural networks

LHC Design	25 Repetitions		50 Repetitions		100 Repetitions	
	Fit	Validation	Fit	Validation	Fit	Validation
25	1	0.995	1	1	1	NaN
50	1	0.9996	1	0.9997	1	1
100	1	0.9999	1	0.9998	1	0.9999
200	1	0.9999	1	1	1	1
400	1	1	1	1	1	1
800	1	1	1	1	1	1
1600	1	1	1	1	1	1

predicted plots show that the points lie along the line, representing a good fit. The residual by predicted plots show that the data is within about 10 *lb* for almost all the data points. The plots for the rest of the neural networks are included in Appendix D Section D.4.1.

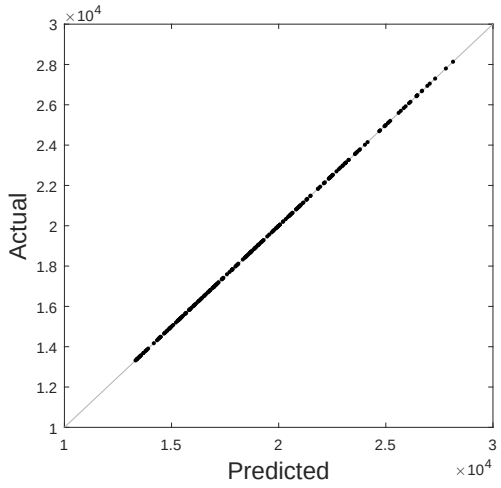
3.7.6.2 Polynomial Regression

Polynomial regressions, or response surfaces, were fit to the data. It should be noted that for LHC designs with very few cases, the goodness of fit metrics can be deceiving. If the number of terms included in the polynomial equation is equal to the number of data points available, a perfect fit can result. This fit, however, is only perfect with regard to the data points being used to create the model. It does not in any way indicate a good fit for the design space of interest. If there are too few points available, no model was fit and the metric is represented as "n/a".

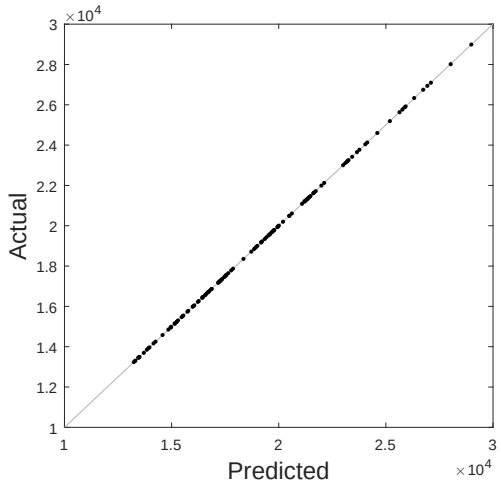
The goodness of fit metrics are shown in Tables 23 and 24. The R^2 values indicate good fits, especially with higher number of cases. The RMSE values are reasonable for higher number of cases as well. The actual by predicted and residual by predicted for the polynomial regressions are included in Appendix D Section D.4.2.

3.7.6.3 Kriging

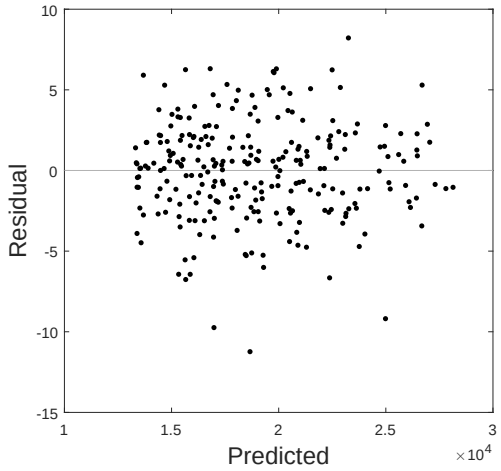
The goodness of fit metrics for the kriging models are shown in Tables 25 and 26. The R^2 metrics show very good fits, although when calculated using the validation data



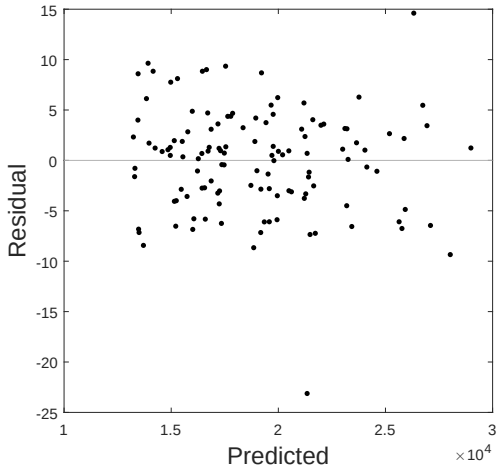
(a) Fit Cases



(b) Validation Cases



(c) Fit Cases



(d) Validation Cases

Figure 76: Goodness of fit plots for neural networks using 1600 cases and 100 required repetitions

Table 23: RMSE values for the polynomial regressions

LHC Design	25 Repetitions		50 Repetitions		100 Repetitions	
	Fit	Validation	Fit	Validation	Fit	Validation
25	69.08	2107.47	66.54	183.21	n/a	n/a
50	78.02	2502.17	127.41	869.45	0.28	375.3
100	24.69	170.03	38.13	948.36	39.18	855.72
200	33.58	95.05	10.89	24.49	47.96	123.52
400	68.4	115.48	17.06	35.95	3.97	26.35
800	50.01	76.63	18.1	28.23	6.33	11.45
1600	136.56	135.58	36.83	53.53	16.07	27.55

Table 24: R^2 values for the polynomial regressions

LHC Design	25 Repetitions		50 Repetitions		100 Repetitions	
	Fit	Validation	Fit	Validation	Fit	Validation
25	1	0.84	0.9996	0.99	n/a	n/a
50	0.9998	0.7504	0.9988	0.8926	1	0.9863
100	1	0.9989	0.9999	0.9446	0.9998	0.9739
200	1	0.9996	1	1	0.9998	0.9992
400	0.9998	0.9994	1	0.9999	1	0.9999
800	0.9999	0.9998	1	1	1	1
1600	0.9993	0.9993	0.9999	0.9999	1	1

Table 25: RMSE values for the kriging models

LHC Design	25 Repetitions		50 Repetitions		100 Repetitions	
	Fit	Validation	Fit	Validation	Fit	Validation
25	55.01	1969.39	3.12	334.34	0.58	NaN
50	30.32	758.52	10.94	149.12	8.01	159.72
100	24.43	58.03	17.2	72.57	15.91	250.43
200	22.26	66.8	16.54	46.99	18.66	34.35
400	18.41	42.9	15.48	29.09	16.7	48.22
800	16.83	29.44	13.51	22.91	10.9	15.62
1600	14.26	17.68	11.42	16.47	9.73	18.42

there are some poor fits. The RMSE values indicate good fits as well. The trend with respect to the number of required repetitions seen earlier with the neural networks is present in these models as well. Consider the models fit using the LHC design with 400 cases. The RMSE using the fit data generally decreases as the required number of repetitions increases. However, the validation data RMSE decreases between 25 and 50 required repetitions and then increases with 100 required repetitions. A similar trend is seen when the LHC design with 1600 cases is used. The kriging model fit using the LHC design with 25 cases and 50 required repetitions is an example of why validation is so necessary. The RMSE and R^2 values are excellent when the fit data is used. When validation data is considered, it is seen that the model is actually quite poor. The actual by predicted and residual by predicted plots are included in Appendix D Section D.4.3.

Table 26: R^2 values for the kriging models

LHC Design	25 Repetitions		50 Repetitions		100 Repetitions	
	Fit	Validation	Fit	Validation	Fit	Validation
25	0.9998	0.8577	1	0.9666	1	NaN
50	1	0.9771	1	0.9968	1	0.9975
100	1	0.9999	1	0.9997	1	0.9978
200	1	0.9998	1	0.9998	1	0.9999
400	1	0.9999	1	0.9999	1	0.9998
800	1	1	1	1	1	1
1600	1	1	1	1	1	1

Table 27: RMSE values for the radial basis functions

LHC Design	25 Repetitions		50 Repetitions		100 Repetitions	
	Fit	Validation	Fit	Validation	Fit	Validation
25	0	9545.3	0	12323.97	0	NaN
50	0	5731.48	0	15337.07	0	12069.2
100	0	5246.55	0	14075.97	0	55710.37
200	0	2565.04	0	2673.44	0	20629.56
400	0	1584.43	0	2270.49	0	3537.94
800	0	866.61	0	512.99	0	1160.51
1600	0.12	380.7	0.11	355.66	0.14	406.68

3.7.6.4 Radial Basis Functions

The goodness of fit metrics for the radial basis functions are included in Tables 27 and 28. It is clear that the fits are very poor. The RMSE and R^2 values when the fit data is used indicates good fits, but the validation metrics prove otherwise. Some of the values of R^2 are significantly negative. In these cases, the fit is so poor that simply using the mean of the performance data as an approximation across the entire design space would result in a better fit. This is evident from considering the definition of R^2, given in Equation 91 in Section 3.7.2. It should be noted that the RMSE values of 0 are values that round to 0, not identically 0. There may be further analysis possible to determine the reason why the fits are so poor or to generate better fits, but that is not necessary for this thesis as good fits have been found using other techniques. Because the fits are so poor, no plots are included.

Table 28: R^2 values for the radial basis functions

LHC Design	25 Repetitions		50 Repetitions		100 Repetitions	
	Fit	Validation	Fit	Validation	Fit	Validation
25	1	-2.3428	1	-44.3442	1	NaN
50	1	-0.3094	1	-32.4296	1	-13.2153
100	1	-0.0387	1	-11.2106	1	-109.4242
200	1	0.7393	1	0.4967	1	-21.9029
400	1	0.8959	1	0.6759	1	0.092
800	1	0.9713	1	0.9879	1	0.9229
1600	1	0.9945	1	0.9942	1	0.9908

Table 29: RMSE values for the support vector regressions

LHC Design	25 Repetitions		50 Repetitions		100 Repetitions	
	Fit	Validation	Fit	Validation	Fit	Validation
25	1573.18	5486.88	1889.5	1765.63	939.27	NaN
50	1490.16	3707.02	875.16	2940.19	1167.68	4098.33
100	1535.61	3590.13	802.69	4690.82	1379.09	5376.72
200	1677.82	3264.46	1241.45	2268.8	750.41	3057.61
400	1786.86	2560.05	1391.62	1883.6	1061.62	2443.6
800	1706.06	2074.36	1606.64	1996.54	1249.67	1943.1
1600	1517.71	1724.71	1417.02	1734.33	1269.42	1819.72

3.7.6.5 Support Vector Regression

The goodness of fit metrics for the support vector regressions are shown in Tables 29 and 30. As with radial basis functions, the fits are very poor. The RMSE and R^2 values calculated using both the fit and validation data indicate poor approximations of the data. The lowest RMSE value is about 1000 and the highest R^2 for fit data, which is about 0.95, corresponds to a very poor validation R^2. Because good fits are already available, no further analysis is performed. No plots are included for these fits.

3.7.6.6 Validation using all Data

The analysis of the surrogate models in the previous sections show the results that would be available if the LHC designs were limited to the corresponding number of cases. For example, the results from the LHC design of 100 cases are based on

Table 30: R^2 values for the support vector regressions

LHC Design	25 Repetitions		50 Repetitions		100 Repetitions	
	Fit	Validation	Fit	Validation	Fit	Validation
25	0.8743	-0.1045	0.7115	0.0693	0.9459	NaN
50	0.9121	0.4522	0.942	-0.2286	0.9038	-0.6391
100	0.9033	0.5137	0.9502	-0.3561	0.806	-0.0286
200	0.8827	0.5778	0.8977	0.6375	0.9526	0.4969
400	0.8665	0.7281	0.898	0.777	0.9243	0.5669
800	0.8844	0.8358	0.8784	0.8173	0.91	0.7837
1600	0.9092	0.887	0.9065	0.8626	0.9076	0.8157

100 cases, for both fit and validation metrics. This would be the case when the methodology is implemented. In this thesis, however, there are an additional 1500 cases available. These can be used as validation cases to confirm the fits approximate the data. Tables 31, 32, and 33 show the validation metrics using all the data from the LHC design with 1600 cases for the neural networks, polynomial regressions, and kriging models respectively. The other model types are excluded due to poor performance. The values in these tables for the LHC designs of 1600 cases are similar, but not equal, to the values in the previous tables. This is because in this case all the data is used, including data used to fit the model. Previously, the data used to fit the model was excluded for the calculation of the validation metric.

The validation metrics using all the data show that the surrogate models provide good approximations of the data. For example, the neural network using 400 cases and requiring 25 repetitions has a global RMSE of 14.73 and R^2 value of nearly 1. This fit was generated using an LHC design of merely 400 points, which resulted in 208 points of data to fit to and 103 points to validate with (from Table 20). The validation metrics quoted here result when the model is tested against all the available data, which consists 1224 points (from Table 19). The model was created with no knowledge of this additional data. It follows that the model is equally accurate across the rest of the design space. This highlights the capability of a surrogate model to approximate the entire design space.

Table 31: Validation metrics for neural networks using all data available

LHC Design	25 Repetitions		50 Repetitions		100 Repetitions	
	RMSE	R^2	RMSE	R^2	RMSE	R^2
25	476.58	0.9854	1190.17	0.907	3594.5	0.0454
50	310.11	0.9938	185.65	0.9977	262.76	0.9949
100	89.28	0.9995	106.31	0.9993	140.31	0.9985
200	36.96	0.9999	51.23	0.9998	83.03	0.9995
400	14.73	1	27.02	1	30.45	0.9999
800	16.19	1	10.45	1	8.37	1
1600	8.49	1	5.43	1	3.83	1

Table 32: Validation metrics for polynomial regressions using all data available

LHC Design	25 Repetitions		50 Repetitions		100 Repetitions	
	RMSE	R^2	RMSE	R^2	RMSE	R^2
25	576.82	0.98	n/a	n/a	n/a	n/a
50	5871.05	-1.2108	2177.62	0.6886	686.09	0.9652
100	1032.69	0.9316	2071.7	0.7181	1822.05	0.7547
200	87.59	0.9995	60.4	0.9998	1133.74	0.905
400	67.94	0.9997	47.78	0.9999	80.43	0.9995
800	59.74	0.9998	47.39	0.9999	14.52	1
1600	52.12	0.9998	29.39	0.9999	9.95	1

Table 33: Validation metrics for kriging models using all data available

LHC Design	25 Repetitions		50 Repetitions		100 Repetitions	
	RMSE	R^2	RMSE	R^2	RMSE	R^2
25	506.12	0.9836	4003.73	-0.0527	4489.55	-0.4892
50	233.58	0.9965	660.17	0.9714	384.15	0.9891
100	118.45	0.9991	183	0.9978	247.35	0.9955
200	58.55	0.9998	63.47	0.9997	86.27	0.9995
400	34.9	0.9999	55.88	0.9998	37.97	0.9999
800	22.63	1	36.95	0.9999	16.88	1
1600	13.31	1	11.09	1	13.28	1

3.7.7 Answer to Research Question 7

The answer to Research Question 7 is determined by examining the goodness of fit metrics for the different surrogate models. Radial basis functions and support vector regressions yield poor fits, and are not included in this discussion. The polynomial regressions are worse than the neural networks and kriging models in most cases. Neural networks and kriging models result in similar metrics, but overall neural networks provide better approximations of the data, as predicted in Hypothesis 7. This is especially evident when the validation using all the data is considered. The answer to Research Question 7 is that neural networks should be used to fit the data for this problem.

3.8 Statistical Methods Conclusion

This chapter focused on the statistical methods available and necessary to achieve the research objective of finding an analytical relationship between launch vehicle parameters and performance. The motivation for using EVT was presented, along with a corresponding research question, repeated below.

Research Question 3 - Does EVT apply to optimization problems?

Research Question 3 was answered using an experiment, where it was shown the EVT does apply to optimization through random searches. When applied to an objective function, a random search is analogous the sample from a population distribution, and the maximum or minimum value is analogous the sample maxima or minima. The application of EVT to trajectory optimization when a random search

was coupled with direct optimization led to Research Question 4.

Research Question 4 - Does EVT apply to the launch vehicle optimization problem when the random global search is coupled with a local direct method?

Research Question 4 was subdivided into two questions, repeated below.

Research Question 4.1 - Is the underlying population of performances for a given vehicle in the domain of attraction of the GEV distribution?

Research Question 4.2 - Is the distribution of results from repeated direct optimization analyses for a vehicle analogous to the distribution of sample maxima?

Research Question 4.1 was answered affirmatively with an experiment. The underlying population of performances for each of the representative vehicles was in the domain of attraction of the GEV distribution. The answer to Research Question 4.2 was not as straight forward. Repeated optimization analyses are not analogous to sample maxima. This was seen clearly in Section 3.4.7. However, after applying various other methods to quantify the difference, it was concluded that repeated optimization analyses were a very good approximation of the distribution of sample maxima and were accepted as such for this thesis.

Given that statistical methods are being used, it is important to measure the confidence associated with the estimates. The bootstrap method was identified as an option for quantifying confidence in this method, leading to the next research question.

Research Question 5 - Does bootstrapping provide an accurate measure of confidence in the performance metric?

Research Question 5 was answered affirmatively and the bootstrap method was included in the methodology being proposed in this thesis.

At this point, a method for evaluating the performance of each vehicle and measuring the confidence of that evaluation had been developed. To generate a surrogate model, this method needed to be implemented to evaluate a set of vehicles to generate the necessary data. Determining this set of vehicles was the subject of the next research question.

Research Question 6 - How should the alternatives to be evaluated be selected?

Design of experiments deals with exactly this kind of problem. Given the depth of literature and examples using design of experiments, this research question was answered without an experiment. It was concluded that LHC designs would be used based on their use in similar problems and their characteristic flexibility for problems where the underlying physics may be nonlinear.

Finally, after a set of vehicles was generated and evaluated, a surrogate model could be fit. Choosing which type of surrogate model to use led to the final research question.

Research Question 7 - Which surrogate modeling technique should be used for launch vehicle performance data?

Based on the metrics used to evaluate the surrogate models, the neural networks provided a better estimate of the data than the other model types considered. While this answer holds for this application and the set of data used to generate the fit, it is not necessarily true for any launch vehicle trajectory optimization problem.

The development of a surrogate model represents a significant contribution to the field of aerospace. The ability to evaluate launch vehicle performance essentially on the fly is of great benefit to launch vehicle designers. This type of analysis enables real-time answers to the "what-if" questions that so often come up. Instead of taking several days or weeks to get answers, decision makers will be able to resolve the issue and move on to the next part of the design.

The main contribution, however, will be the methodology to develop the surrogate model, shown in Figure 73. A surrogate model in itself is useful for the vehicle considered. Having a method to develop surrogate models is useful for future programs as new launch vehicle concepts are explored. This means a designer can develop a surrogate model for the problem of interest and provide incredible analysis capability in conceptual design.

This chapter coupled with Chapter 2 has proposed a method to achieve the research goal and developed some research questions along the way. The next chapter

will apply the method to an extended design study and showcase some example applications.

CHAPTER IV

RAPTOR METHODOLOGY IMPLEMENTATION AND APPLICATIONS

Throughout the last two chapters, a set of research questions regarding the methodology proposed in this thesis have been addressed. By the conclusion of Chapter 3, each element of the proposed methodology had been addressed either from the literature or through experimentation. This complete methodology, shown in Figure 73, is termed the Rapid Trajectory Optimization Routine, or RAPTOR. Figure 77 shows the RAPTOR methodology with each of the research questions mapped to the corresponding elements of the methodology. The research questions are listed below.

1. Which optimization technique should be used to generate the performance data?
2. 2.1. How should the control structure for a launch vehicle trajectory optimization problem be selected?

   2.2. How can this method of selecting control structures be applied to an entire design space?
3. Does EVT apply to optimization problems?
4. Does EVT apply to the launch vehicle optimization problem when the random global search is coupled with a local direct method?

   4.1. Is the underlying population of performances for a given vehicle in the domain of attraction of the GEV distribution?

   4.2. Is the distribution of results from repeated direct optimization analyses for a vehicle analogous to the distribution of sample maxima?

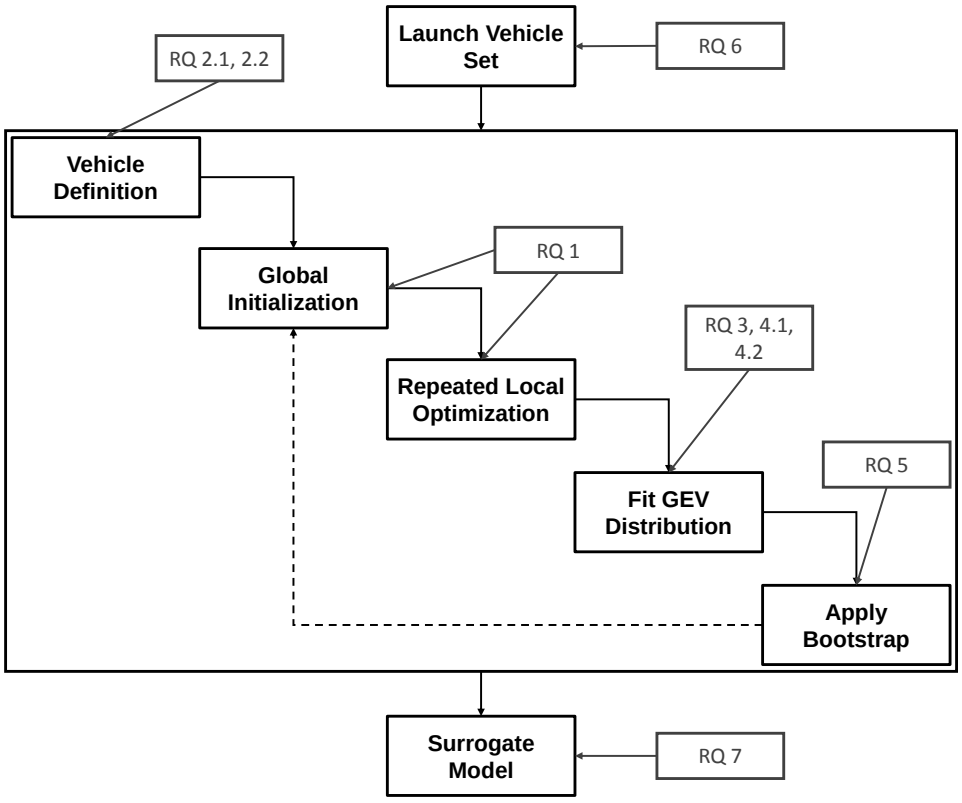


Figure 77: RAPTOR methodology with research questions mapped to individual elements

5. Does bootstrapping provide an accurate measure of confidence in the performance metric?
6. How should the alternatives to be evaluated be selected?
7. Which surrogate modeling technique should be used for launch vehicle performance data?

In this chapter, RAPTOR is implemented in a design study where the design variable bounds are extended beyond the ranges used in Chapters 2 and 3. This study is then compared to the results if current methods were implemented. In the final section of this chapter, several use cases are presented for a surrogate model resulting from RAPTOR.

Table 34: Design variable ranges for extended design study

Variable	Abbreviation	Dimension	Max	Min
Vehicle TW	G_TW	N/A	1.5	1.1
Upper Stage TW	US_TW	N/A	0.75	0.1
Upper Stage IMF	US_IMF	N/A	0.15	0.03
Core/booster ISP	CB_ISP	s	500	350

4.1 Extended Design Study

As this study is carried out, the specific inputs for each step of the process will be discussed. Figure 78 shows the RAPTOR methodology with the required data inputs at each step in the process. In addition, the sections where each of the main steps are implemented are shown in brackets. This figure will be referred to as the process is implemented.

The extended design study is carried out by increasing the ranges to the design variables given in Table 5 in Section 2.5.2. The extended ranges are given in Table 34. Once the design variables and ranges are chosen, a set of vehicles, designed as an LHC design, can be generated. These vehicle will be used to fit a surrogate model at the end of the RAPTOR methodology.

When a new design study is being conducted, the OEPI method should be implemented to establish how the control parameters will approximate the control function. In this case, the OEPI method has been applied for the same design variables. Consequently, the same control structure will be used. It should be noted that it is assumed that the extended ranges on the design variables will not negatively affect the control structure. Once this is done, a set of initial guesses for the control parameters is selected. This represents the global initialization of the optimization process. The next step is to run local direct optimization for each combination of vehicle and initial guess. In the figure, 1600 vehicles are chosen with 500 initial guesses, resulting in a total of 800,000 individual optimization analyses. The trajectories that do not meet the final conditions are discarded. The GEV distribution is fit to the data points that

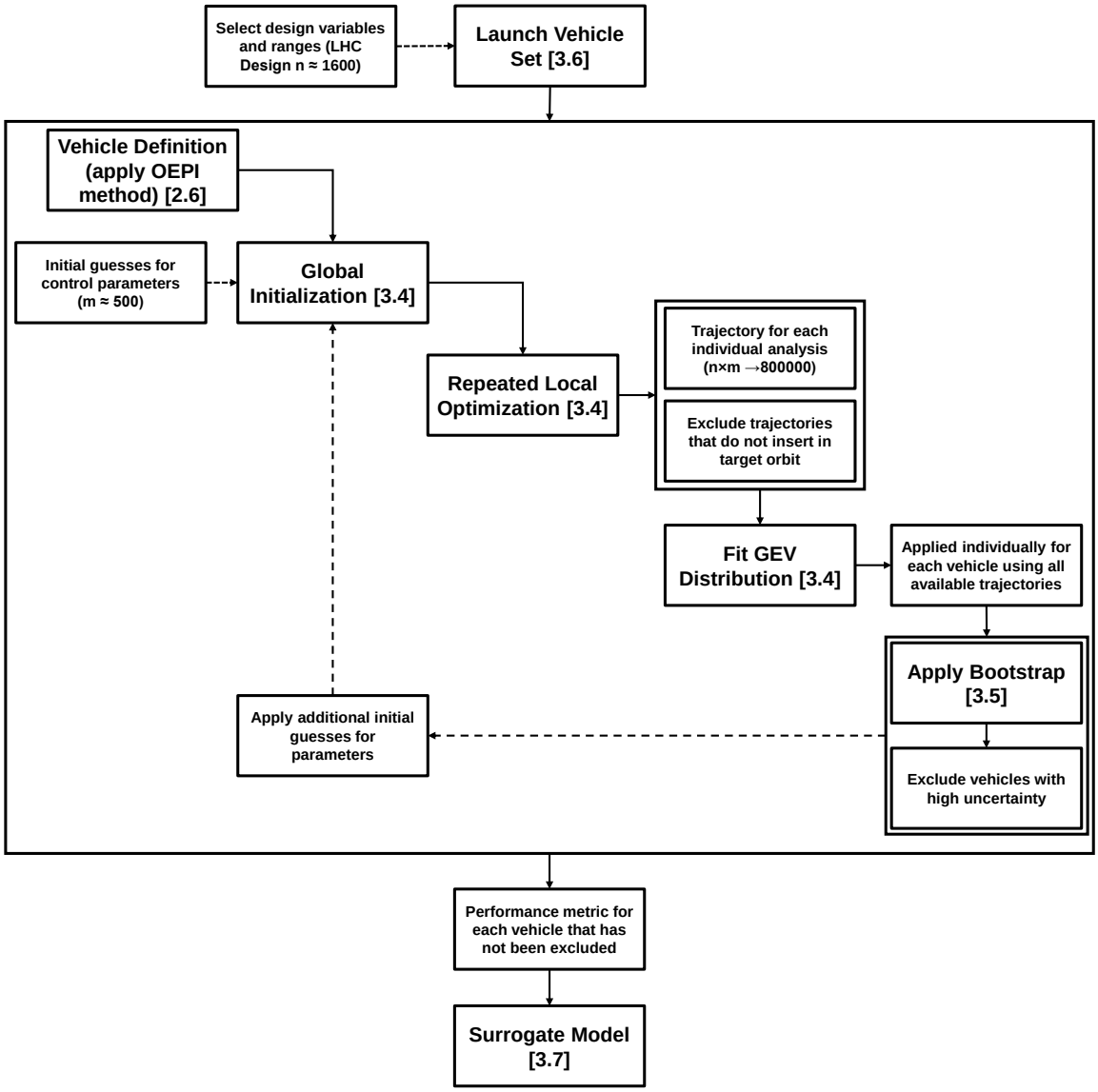


Figure 78: RAPTOR methodology with data inputs at each step

result for each of the vehicles. It should be noted that some vehicles may have no trajectories that insert in the orbit, others may only have one, and others may have a feasible trajectory for each of the initial guesses used. The bootstrap method is applied and any vehicles that do not meet the requirements, as described in Section 3.6.1, are excluded. What results is a set of vehicles each described with a GEV distribution based on repeated trajectory optimization that can be used to generate a performance metric with low uncertainty. This data is then used to generate a surrogate model.

Applying the RAPTOR method (with 750 plus an additional set of 500 initial guesses for the control parameters) resulted in a total of 787 usable vehicles of the original 1600 when at least 25 repetitions are required for each vehicle. This is a significantly lower number than the previous study, which resulted in 1224 usable vehicles. This is directly caused by the larger design space. The extended ranges in Table 34 represent a significant increase in the four dimensional space. If the original design space is thought of as a unit four dimensional volume, the expanded design space covers over 21 unit four dimensional volumes. The same number of vehicles is being used to cover a space over 21 times larger. In addition, the initial guesses for the control parameters may not apply as well to the expanded design space, so fewer will lead to successful trajectories. Finally, there are regions of the design space that do not have any feasible vehicles. This is seen clearly in Figure 79. The design space limits considered are plotted for each design variable. The subplots showing the G_TW variable reveal that no vehicles with a gross thrust-to-weight value of over about 1.4 have feasible trajectories. Some of the features mirror what was seen previously in Figure 74. For example, there is no data for low values of G_TW and US_TW. However, the extended design study reveals similar interactions between G_TW and CB_ISP as well as US_TW and CB_ISP.

The surrogate model provides a decent fit of the design space. Recall that this

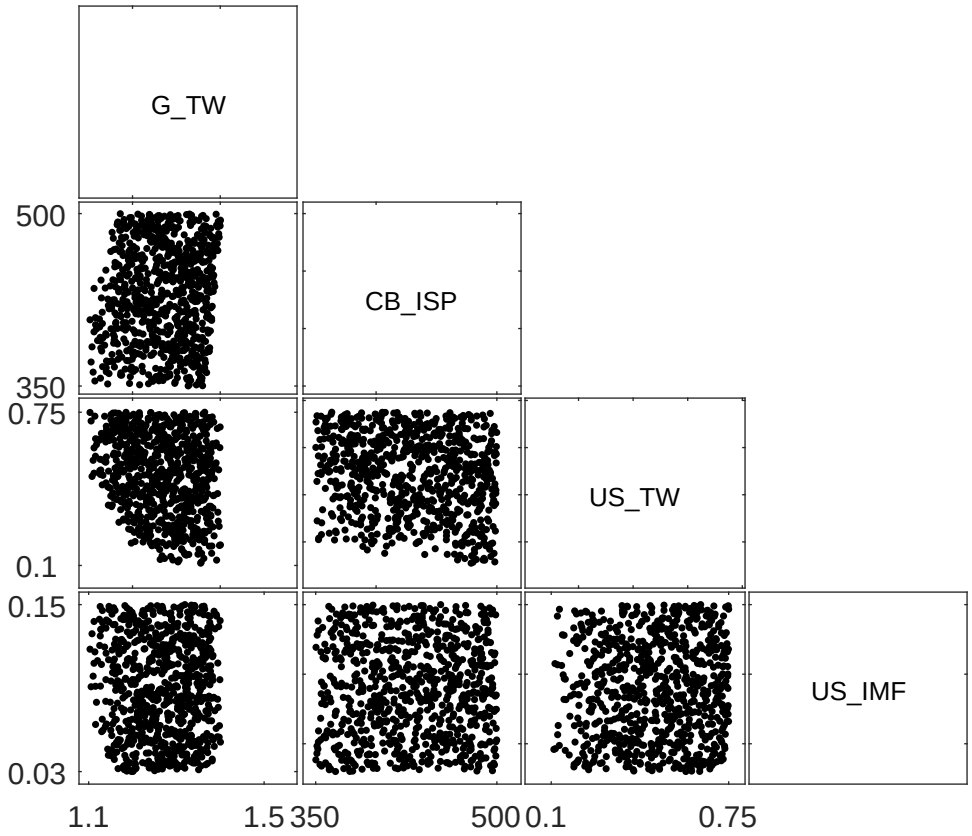


Figure 79: Scatter-plot of feasible vehicles in the extended design study when the number of required repetitions is 25

Table 35: Goodness of fit metrics of the neural network for extended design study

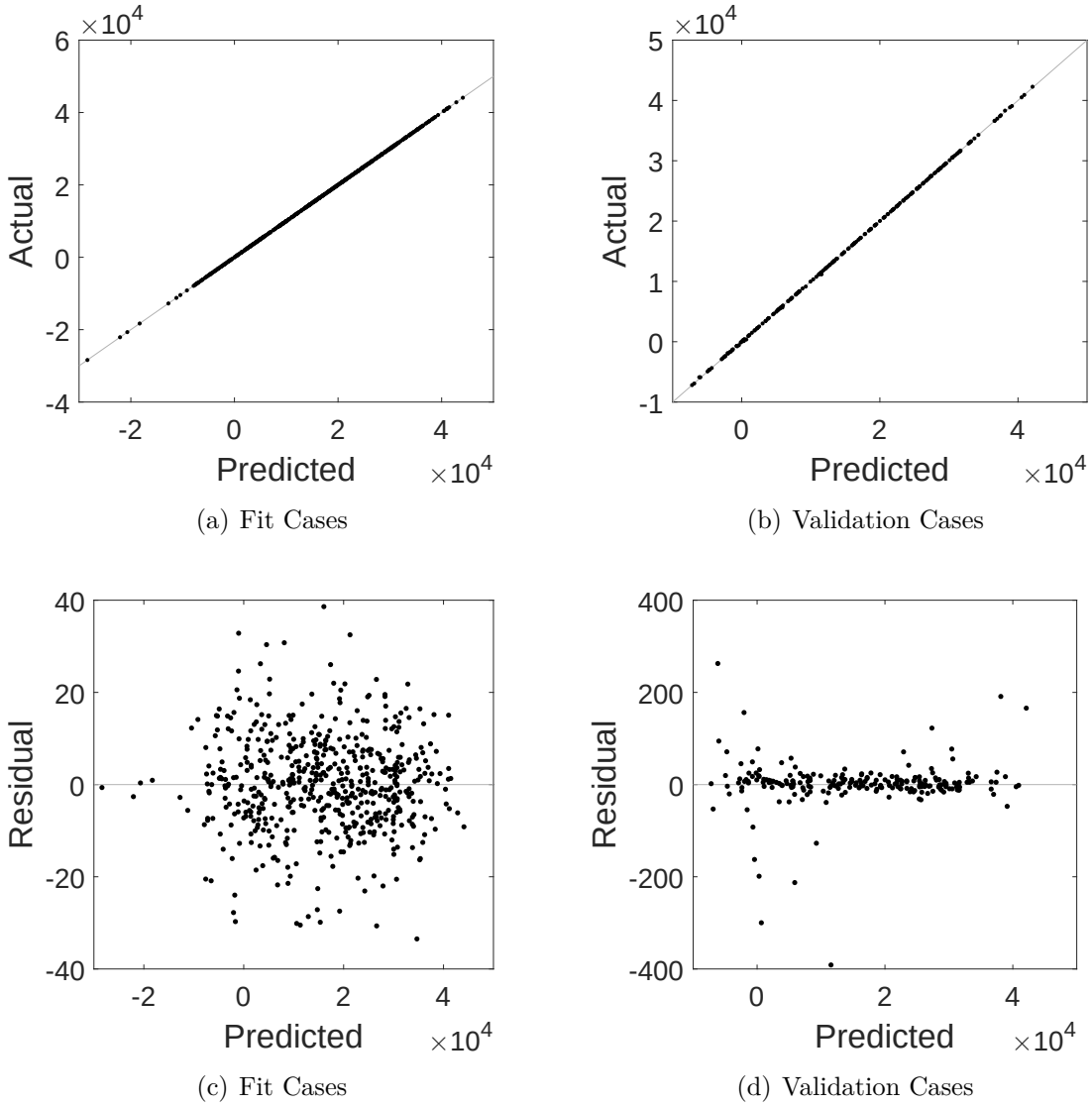
	Fit	Validation
RMSE	10.35	53.82
R^2	1.000	1.000

design space represents a hyper-volume over 21 times greater than the original design space considered in Chapters 2 and 3. As previously, 30% of the data is set aside for validation. This means the model is fit without any knowledge of that data. The goodness of fit metrics are given in table 35. The R^2 values represent very good fits. The RMSE value for the validation data is higher than may be desired. However, it should be noted that the values being estimated are on the order of $10,000$ lb. It is common to normalize the RMSE values, where the RMSE value is divide by the range of the predicted values. In this case the range is over $70,000$ lb, and the normalized RMSE for the validation data is 0.07%.

The goodness of fit plots for the neural network are shown in Figure 80. The actual by predicted plots (Figures 80(a) and 80(b)) reveal that the data is well predicted by the model. The reference line represents a perfect fit. The residual by predicted plot for the fit data reveals a good fit. The analogous plot for the validation data shows that there is some error in the model, especially in the upper and lower extremes. However, for the most part, the points are within around 50 lb of the data.

4.1.1 Reducing the Design Space

The effect of reducing the design space will be considered in this section. In Chapters 2 and 3, a design space was considered to develop the RAPTOR methodology. In this section, the RAPTOR methodology was implemented to consider a larger design space. The effects on the surrogate model fits will be tracked in order to consider the scalability of this problem. The design space reduction will be conducted by discretely moving the bounds of the design space given in Table 34 to those given in Table 5. The bounds will be reduced linearly by 25%, 50%, 75%, and 100% for a total



(a) Fit Cases

(b) Validation Cases

(c) Fit Cases

(d) Validation Cases

Figure 80: Goodness of fit plots of neural networks for the extended design study

Table 36: Reduced design space ranges

Design Space	G_TW		US_TW		US_IMF		CB_ISP	
	Max	Min	Max	Min	Max	Min	Max	Min
Expanded	1.5	1.1	0.75	0.1	0.15	0.03	500	350
Reduce by 25%	1.4625	1.115	0.6875	0.1175	0.1425	0.035	487.5	362.5
Reduce by 50%	1.425	1.13	0.625	0.135	0.135	0.04	475	375
Reduce by 75%	1.3875	1.145	0.5625	0.1525	0.1275	0.045	462.5	387.5
Reduce by 100%	1.35	1.16	0.5	0.17	0.12	0.05	450	400

of four reduced design spaces, including the original. These are shown in Table 36. The design space reduced by 100% will be equivalent to the original, given in Table 5.

As the design space is reduced, the number of available points will be accordingly reduced. Table 37 gives the number of cases available for fitting and validating the models for each of the design spaces (30% of the cases are used to validate the models). In addition, a hyper-volume factor is reported. In two dimensions, a design space area can be calculated by multiplying the ranges considered for each of the dimensions. Similarly, in four dimensions the hyper-volume can be calculated. The original design space is considered to be a unit hyper-volume, and the hyper-volume factor represents how many unit hyper-volumes are contained within the respective design spaces. In two dimensions, doubling the ranges in each dimensions represents a fourfold increase in the volume. As more dimensions are considered, even small increases in the range can drastically increase the amount of space considered. This hyper-volume factor should be taken into account as the results of this design study are presented. In this case, a larger space is being considered using the same amount of points and the results reflect that.

Figure 81 shows the feasible vehicles for each of the designs in Table 36. The points shown in purple represent the original design space used in Chapters 2 and 3. Any of the points that are included in a reduced design space are also included in any larger design space. For example, the red points represent the design space that

Table 37: Number of cases and hyper-volume factor for reduced design spaces

Design Space	Fit Cases	Validation Cases	Hyper-volume Factor
Expanded	565	222	21.3
Reduce by 25%	373	145	12.1
Reduce by 50%	228	89	6.3
Reduce by 75%	124	51	2.8
Reduce by 100%	50	18	1

has been reduced by 50% from the expanded, but these points are also part of the design space reduced by 25% and the expanded space. In some cases the extended design ranges capture more features of the design space. For example, the relationship between CB_ISP and G_TW at high values for the former and low values for the latter is not seen when the original design space was explored. A similar effect is in the extended design study with low values of US_TW and CB_ISP. Additionally, the cut off of designs at high values of GT_TW was not seen previously. Other effects, like the lack of feasible vehicles in the region of the design space characterized by low values of thrust-to-weight, was seen in both design spaces. The value of expanding the design space is to explore areas that may contain promising designs, even if they are on the limit of the feasible regions.

A surrogate model can be fit using each of the design spaces shown in Figure 81. These surrogate models are only valid in the areas of the design space containing feasible vehicles. For example, fitting a model using the points in the designs space that has been reduced by 100% could not be applied to any of the regions with points shown in black in Figure 81. Table 38 shows the goodness of fit metrics for the neural networks. These metrics are shown for the data used to fit the model as well as the data used to validate the model. In addition, a column labeled "All" is included. This column contains the metric values when all the data, including the data used in Chapters 2 and 3, is included. This represents an additional 1224 cases that are being used to quantify the error in the models.

The R^2 values for the models indicate good fits for each of the design spaces. The

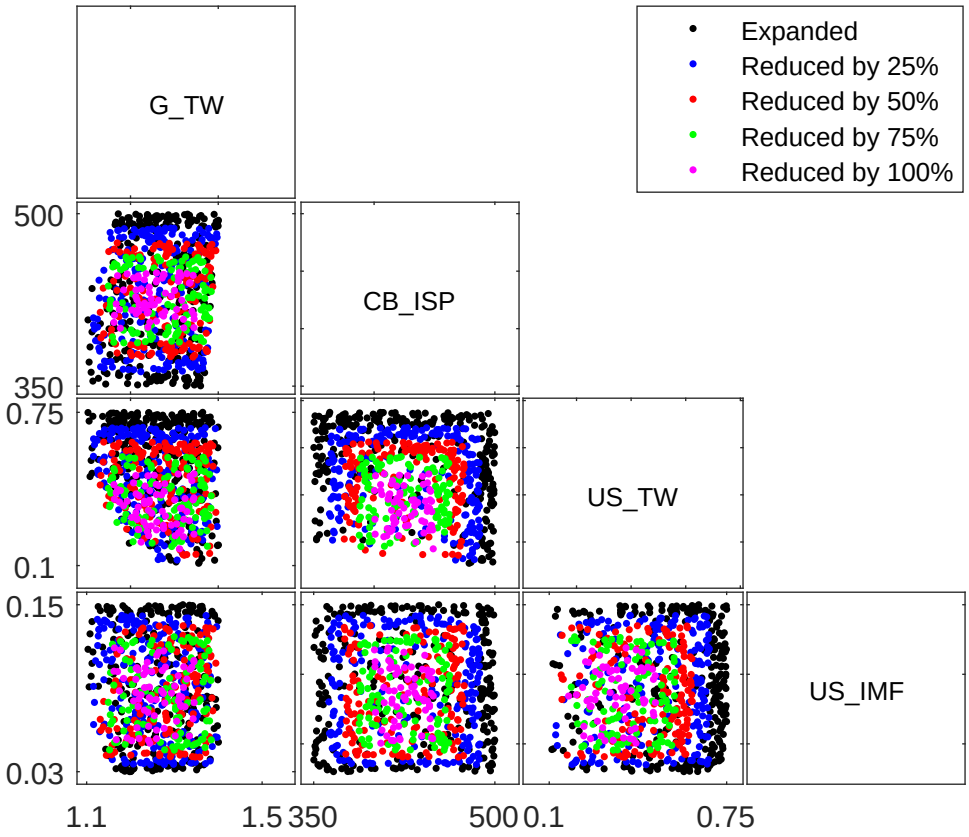


Figure 81: Scatter-plot of feasible vehicles for the reduced design spaces given in Table 36

Table 38: Goodness of fit metrics of the neural networks for the reduced design spaces

Design Space	RMSE			R^2		
	Fit	Validation	All	Fit	Validation	All
Expanded	10.35	53.82	61.42	1	1	1
Reduced by 25%	31.75	83.57	60.58	1	0.9999	0.9999
Reduced by 50%	14.76	83.7	99.84	1	0.9999	0.9997
Reduced by 75%	23.4	81	85.49	1	0.9999	0.9997
Reduced by 100%	4.1	62.41	122.8	1	0.9999	0.9994

RMSE values, however, beg closer inspection. Recall that the expanded design space represents a hyper-volume over 21 times larger than the original space considered. Considering that the same number of vehicles were used to model both spaces, the neural network does a very good job of predicting the performance of each of the vehicles. Similar results are seen for the rest of the models, although the RMSE when all vehicles are considered does tend to increase as the design space is reduced. The design space reduced by 100%, which is equivalent to the original design space, has an RMSE value of 122.8 when all the vehicles are considered. Recall, however, that the values being modeled are on the order of $10,000\ lb$. Additionally, this should be compared to the metrics found when a similar number of points are used to generate the model. In this case, 50 points were used to fit the model and 18 were used for validation. In Section 3.7.6, a model was fit using a similar amount of points when the LHC design of 100 cases was used. That model was generated using 58 points to fit and 23 to validate. The RMSE values for the fit and validation data are 10.27 and 48.1 respectively, as seen in Table 21. The RMSE value when all the points were considered is 89.28, as seen in Table 31. These values are comparable to what are seen here. This is an expected result, as the RAPTOR method has been implemented in both cases with similar amount of points.

This extended design study can be used to make inferences about the scalability of the RAPTOR method. In this thesis, only four design variables are considered in a limited design space. Extending the ranges on the design variables showed that the method can be implemented in a significantly larger design space. As far as the ranges on the design variables are concerned, there is no reason to believe that the RAPTOR methodology limits the vehicles that can be considered. Limitations would arise as the ranges move into areas of the design space where no feasible trajectories for the vehicles exist. For example, the gross thrust-to-weight value is limited to around 1.4. Vehicles with a higher thrust-to-weight do not make it to orbit due

to acceleration limits. (Note: this limit was an assumption made when the vehicle was modeled in Section 2.5. If removed, that area of the design space may contain vehicles). The other factor that should be considered when the design ranges are increased is the computational effort required to evaluate the space. A larger space requires more vehicles to properly model. This is an inherent limitation in design space exploration, and not unique to the RAPTOR methodology.

The four design variables considered in this thesis represent a small sample of the design variables that must be considered when designing a vehicle. The RAPTOR methodology can be applied to any number of design variables, but as with any design space exploration tool, the curse of dimensionality will take its toll. As design variables are added, a larger number of vehicles will be required to model the space with the same density. The number of vehicles required increases exponentially with number of dimensions. If larger computational resources are dedicated to the RAPTOR methodology, however, larger design studies could easily be carried out. It should be noted that the curse of dimensionality exists as the current method for performance evaluation is implemented as well. In the following section, the RAPTOR method will be compared to modified current methods.

4.2 Comparison to Current Practices

The RAPTOR method will be compared to current practices in two different ways. The RAPTOR method uses repetitions to determine a specific performance value for a vehicle. Traditionally this is done by an analyst in a time consuming process the results in a handful of vehicles being considered in a day [176]. The RAPTOR method is able to provide performance metrics at the rate of about 10 case every hour using a Dell OptiPlex 790 with a Core i7 processor, representing a significant speed up. The study done in Chapters 2 and 3 took about 40 hours when implemented using RAPTOR, but would take at least 50 work days for an analyst to implement, assuming

they could evaluate one vehicle every hour during normal working hours. To provide a more direct comparison, the RAPTOR method will be compared with two other ways of automation that could be implemented. The first would be considering the process when the first feasible case for every vehicle is used as the performance value. This would mean running different initial guesses for each vehicle until a feasible solution is found. The second method will be to use a single initial guess for each vehicle and considering that data that results. Both these methods are implemented and the results discussed below.

4.2.1 First Feasible Case

The first feasible case can be used as the performance value for each vehicle. This would represent a drastic decrease in the amount of time required to generate the data. Depending on many initial guesses for the control parameters are required before a feasible trajectory is found, the decrease in computational time could be between two and three orders of magnitude. However, there is a major problem when this approach is used. In Section 3.1.1, it was shown that considering a single trajectory was equivalent to selecting a performance value from a random distribution. The effects of this will be shown in this section. The data from the extended design study in the previous section has been filtered to only contain the first feasible trajectory for each vehicle. Of the 1600 vehicles considered, 1077 had at least one feasible trajectory. Since only one trajectory is being considered, none of the statistical methods discussed in Chapter 3 can be applied. All of these require a set of data. A surrogate model can still be fit to the single data point for each vehicle. The goodness of fit metrics are rather poor, but even if a good surrogate model was created, the disadvantage of this method is seen clearly in a profile plot.

A profile plot represents trends of the predicted values with respect to the design variables when the design variables are set to specific values. In this case, the variables

are set to the baseline values for the Delta IV Heavy, given in Table 5. The profile plots are shown in Figure 82. These plots are designed to represent the partial derivative of the performance metric with respect to the design variables at a specific point in the design space. What is seen here, however, are very erroneous trends, particularly with respect to the thrust-to-weight variables. Given the physics of launch to space, it does not follow that the propellant remaining in orbit would follow a trend seen in Figure 82(c) as the thrust-to-weight of the upper stage changes. This is an artifact of poor data, not the physics of the underlying problem. Similar plots are seen if other points are chosen throughout the design space instead of the baseline.

While using the first feasible trajectory to model the data results in a drastic decrease in computational time, the data results in erroneous trends. Implementing this in a conceptual design study would be of no value.

4.2.2 Single Control Guess

Using a single initial guess for the control parameters is another way to find performance data for each of the vehicles. As with using the first feasible trajectory, this method represents a drastic decrease in computational time. The difference could be greater than two orders of magnitude. The first question when this method is implemented is which initial guess to use for the control guess. This question will be addressed later on in the discussion, and for now the guess that resulted in feasible trajectories for the most vehicles in the extended design study will be used. In this case, a single control guess resulted in feasible trajectories for 782 vehicles. This is merely 5 feasible vehicles fewer than where found when using the RAPTOR method. Plotting the feasible points, shown in Figure 83 reveals that a similar amount of the design space is considered using this single initial guess as compared to using the RAPTOR method (Figure 79).

There are two challenges that result when implementing this method. The first

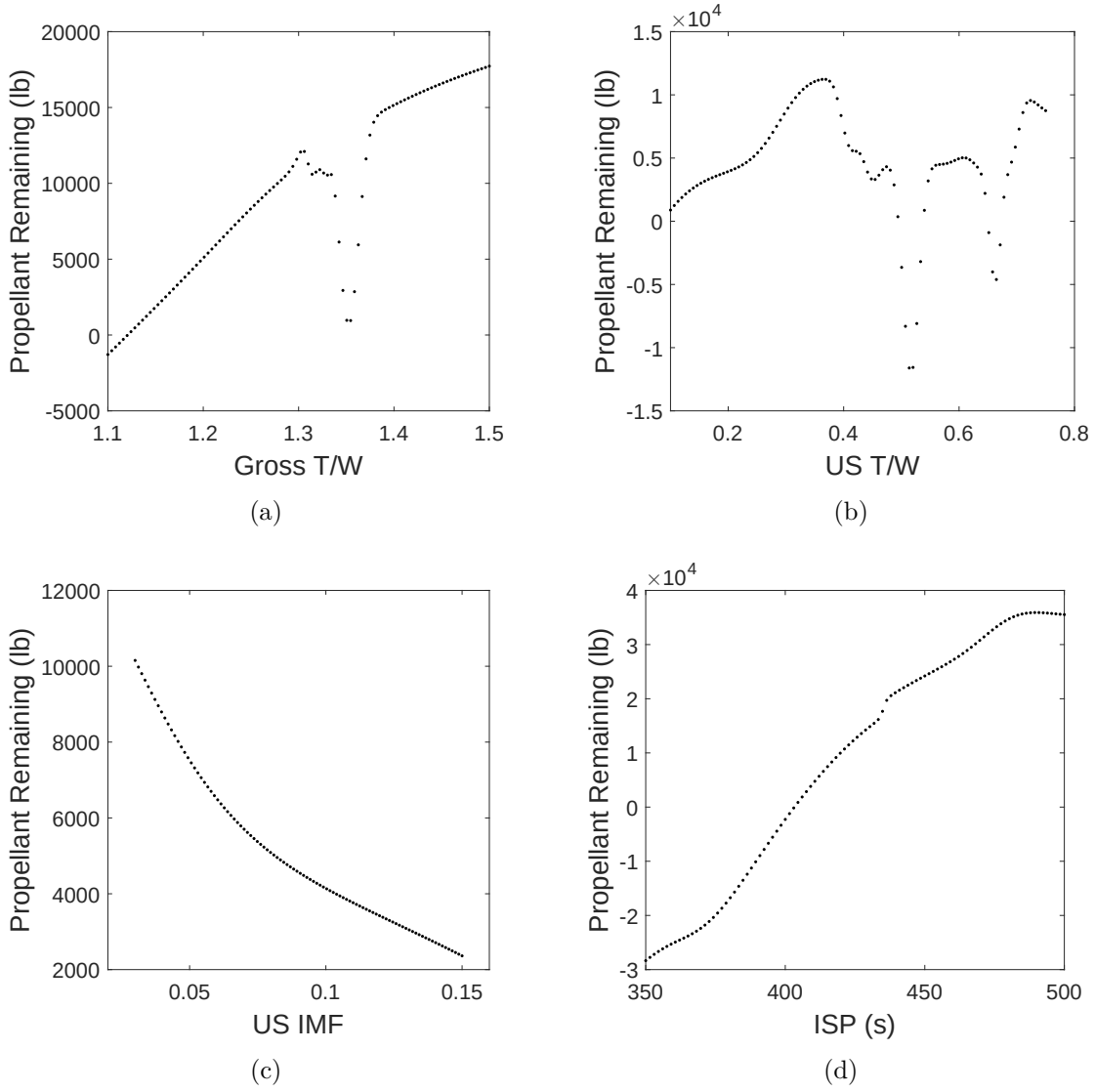


Figure 82: Profile plots for surrogate model fit using the first feasible trajectory for each vehicle

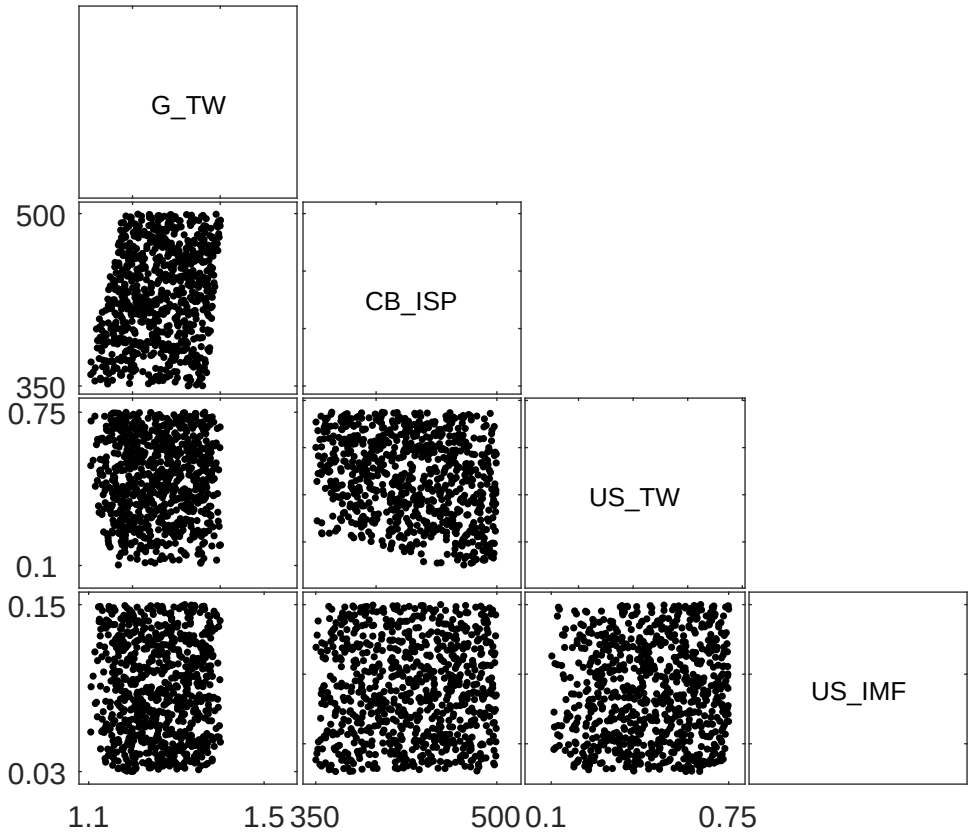


Figure 83: Scatter-plot of feasible vehicles in the extended design study when a single guess is used for the control variables

has been seen previously. When a single initial guess is used, erroneous trends result in the data and consequently in the model. The profile plots using the baseline values given in Table 5 are shown in Figure 84. In this case the erroneous trend is only seen with the gross thrust-to-weight in Figure 84(a). However, these effects are seen again if the trends at points in the design space other than the baseline are considered.

The second challenge with using a single control guess was alluded to previously and consists of choosing the initial guess to use. In this case, the best initial guess was chosen after considering multiple. Implementing this again would defeat the purpose of applying a single initial guess. For the analyses in the previous paragraphs, the best initial guess was chosen. However, this is not known *a priori*. To the best of the author's knowledge, there is no systematic method for selecting initial guesses for a vehicle. In addition, the process of optimizing an initial guess is extremely difficult to predict, and even selecting an initial guess close to an expected final solution does not guarantee that the solution will result. If an initial guess is chosen at random, the number of feasible trajectories that result will likely be a lot fewer. For the extended design study in Section 4.1, 120 vehicles or fewer would result 50% of the time if a random initial guess was chosen. A vehicle count of 222 or more would result less than 25% of the time. Considering this, coupled with the erroneous trends that result in the data, it is concluded that implementing a method that relies on a single initial guess will not result in a usable performance model for conceptual design.

4.3 Sample Applications

The goal of this thesis has been to develop a method for evaluating the performance of a launch vehicle in the context of conceptual design. This objective has been achieved in the RAPTOR methodology. This section explores some of the possible applications of a surrogate model that results from implementing RAPTOR. The Delta IV Heavy has been designed and built, but if this method were applied to a vehicle currently

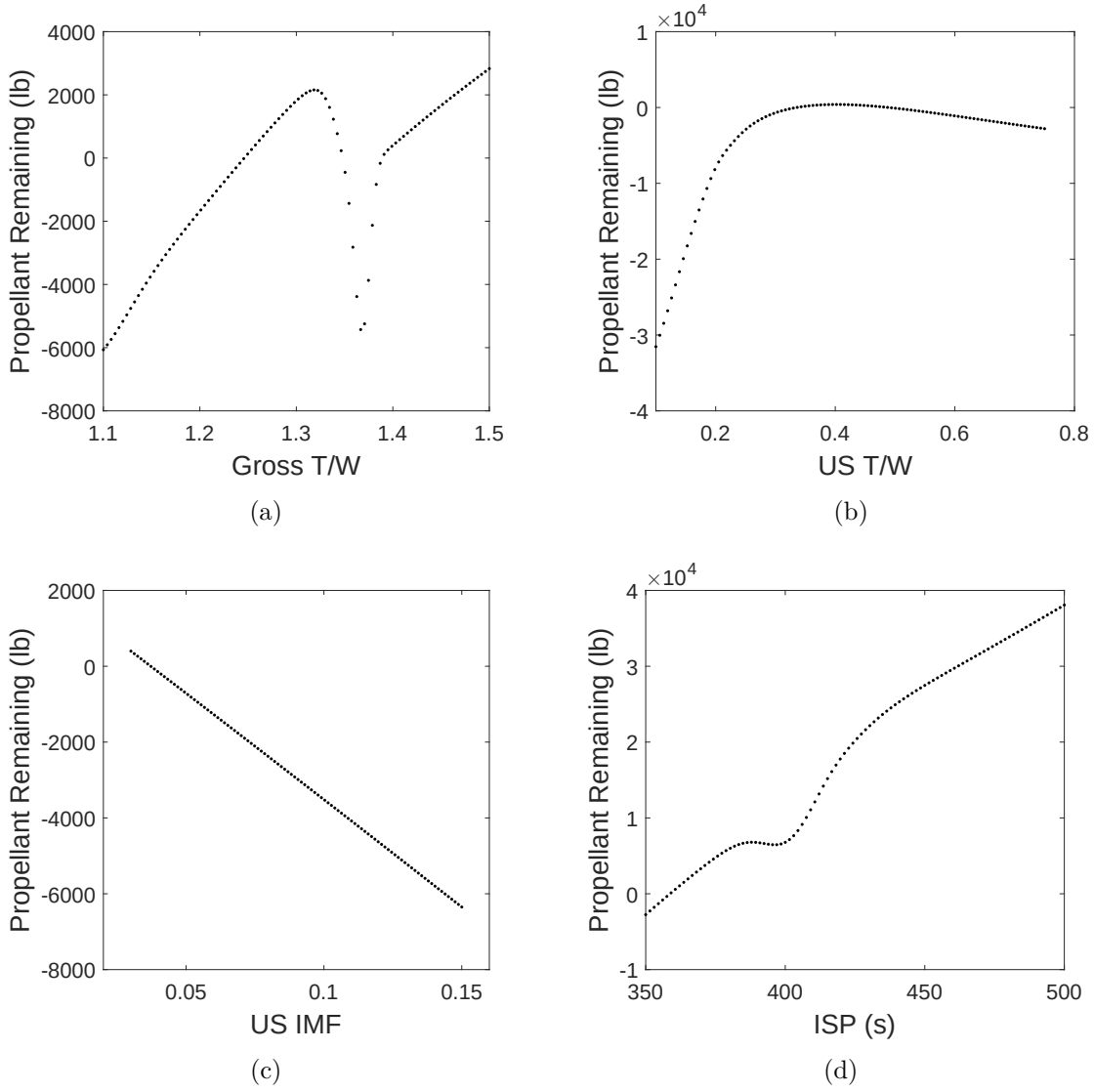


Figure 84: Profile plots for surrogate model fit using a single initial control guess for each vehicle

in conceptual design, some of the studies discussed here could be carried out.

4.3.1 Design Space Exploration

One of the simplest ways to utilize a performance surrogate is to explore the design space. The profile plots, explained previously in Section 4.2, are a useful way of visualizing the space. The profile plots using the baseline values given in Table 5 are shown in Figure 85. These plots are based of the surrogate model described in Section 4.1. It is interesting to compare these profile plots to the ones given in Figures 82 and 84. In those figures, several discrepant trends appeared. Specifically the extreme performance degradation at higher values of gross thrust-to-weight. This trend is seen in Figure 85 as well, but to a lesser extent, and there is no drastic performance recovery. The degradation is due to the limitation on acceleration throughout the vehicle's trajectory. At higher thrust-to-weight the acceleration limit is violated, leading to poor trajectories. It is interesting to note that the core/booster ISP value has the most effect on the performance at this point in the design space. Increasing the upper stage IMF value results in a constant decrease in performance across the design range considered. Performance increases with upper stage thrust-to-weight to a certain point and then slowly degrades. The profile plots shown here are static, but they can be generated virtually instantaneously in a graphical user interface type setting for use in real time decision making.

The point in the design space at which the profile plots are generate can be changed to see various other effects. For example, Figure 86 shows the profile plots can be generated using the point give in Table 39. Some of the trends are similar while others are different. Performance decreases monotonically, for example, with upper stage thrust-to-weight. This may seem counter intuitive, but it is a result of the upper stage trajectory. In these simulations the upper stage burns continuously until reaching the target orbit. At high thrust-to-weight values, more fuel is being

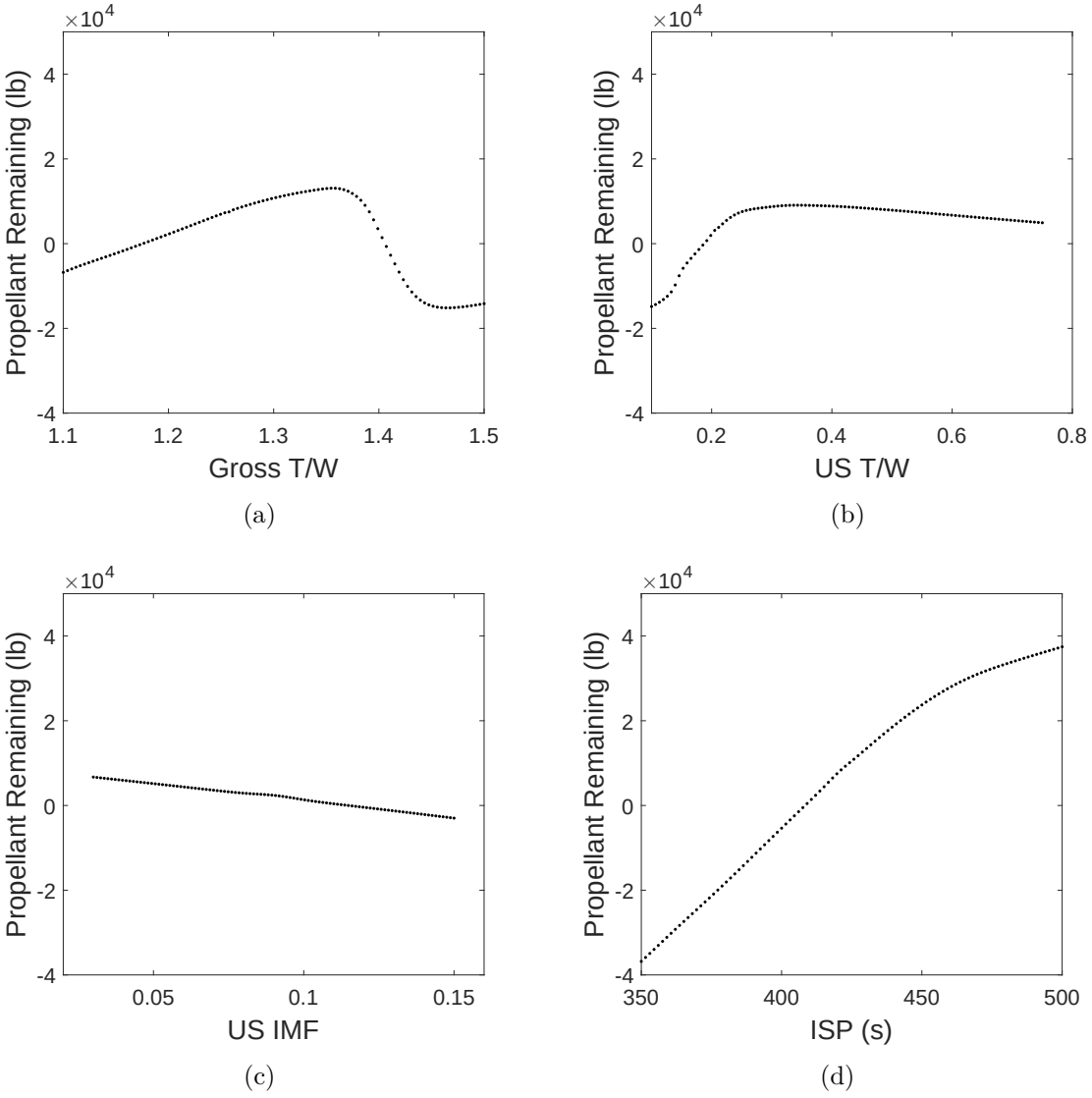


Figure 85: Profile plots for surrogate model from extended design study using Delta IV baseline values

Table 39: Design point for profile plots in Figure 86

G_TW	US_TW	US_IMF	CB_ISP
1.35	0.45	0.05	485

consumed every second than a lower thrust-to-weight values for a given ISP. The lower thrust-to-weight allows the upper stage to perform a more efficient trajectory which takes more time, but actually uses less propellant. This is because the gravity losses for the trajectory are minimized. Figure 87 shows the trajectories for Vehicles 10 and 12 from the set of representative vehicles in Table 6. These vehicles are identical except for the upper stage thrust-to-weight. Vehicle 10 has a lower upper stage thrust-to-weight than Vehicle 12. The optimal trajectory for Vehicle 12 results in more gravity losses, which occur when thrust is being used to counteract gravity instead of increase velocity. This is seen most clearly in the differences in the altitude profile in Figure 87(b). The difference in performance, or propellant remaining, is seen in the weight profile in Figure 87(c). The profiles are virtually identical until the upper stage is ignited. At that point, Vehicle 10 experiences a slower rate of propellant consumption. At orbit insertion, Vehicle 10 has more propellant remaining than Vehicle 12.

This example begs two questions. The first is why Vehicle 12 did not simply fly the same trajectory as Vehicle 10 up until the staging point. If the velocity profile is examined, it can be seen that the vehicles ignite the upper stage at different velocities. The answer to this is that given the thrust value for Vehicle 12, the trajectory would over shoot the target orbit if the upper stage ignited at the same velocity as Vehicle 10. It would either have too much velocity at the desired altitude or be too little altitude at the desired velocity. The second question is why not simply throttle the upper stage engine. This is a perfect example of how the analysis provided by RAPTOR can be useful. Throttling the upper stage engine was not included as a design variable in this analysis. Considering the results, however, perhaps it should have been. If a

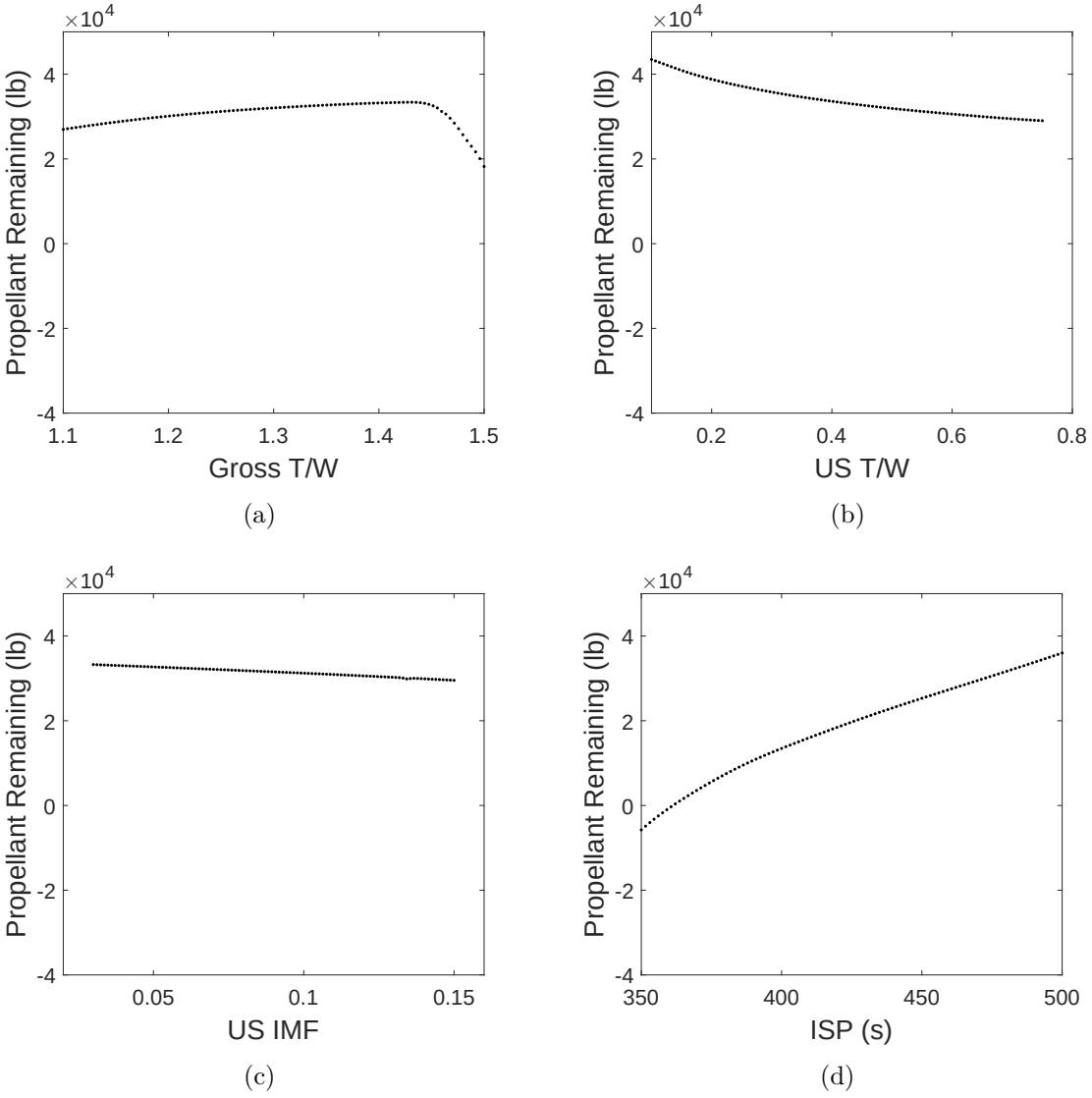


Figure 86: Profile plots for surrogate model from extended design study using values in Table 39

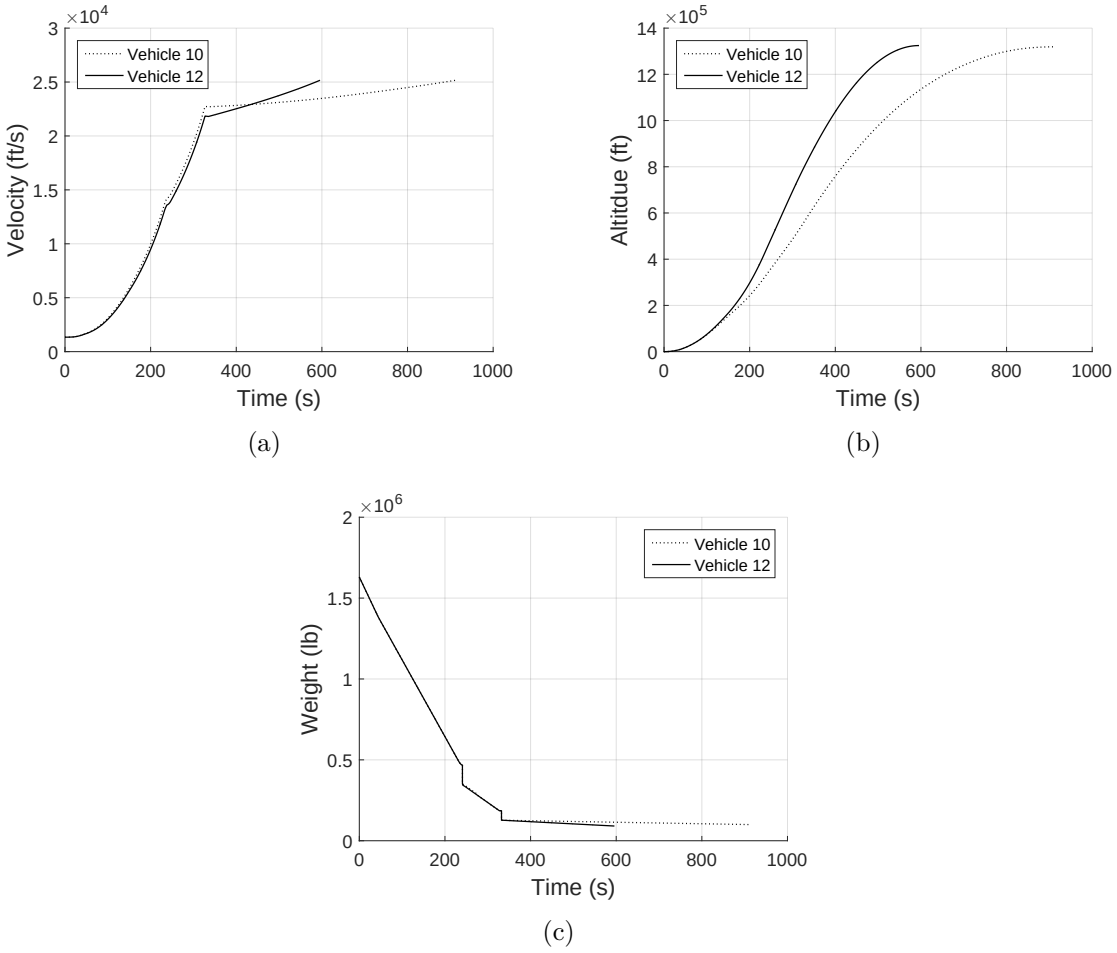


Figure 87: Comparison of vehicle trajectories to isolate effect of upper stage thrust-to-weight

Table 40: Optimized design point

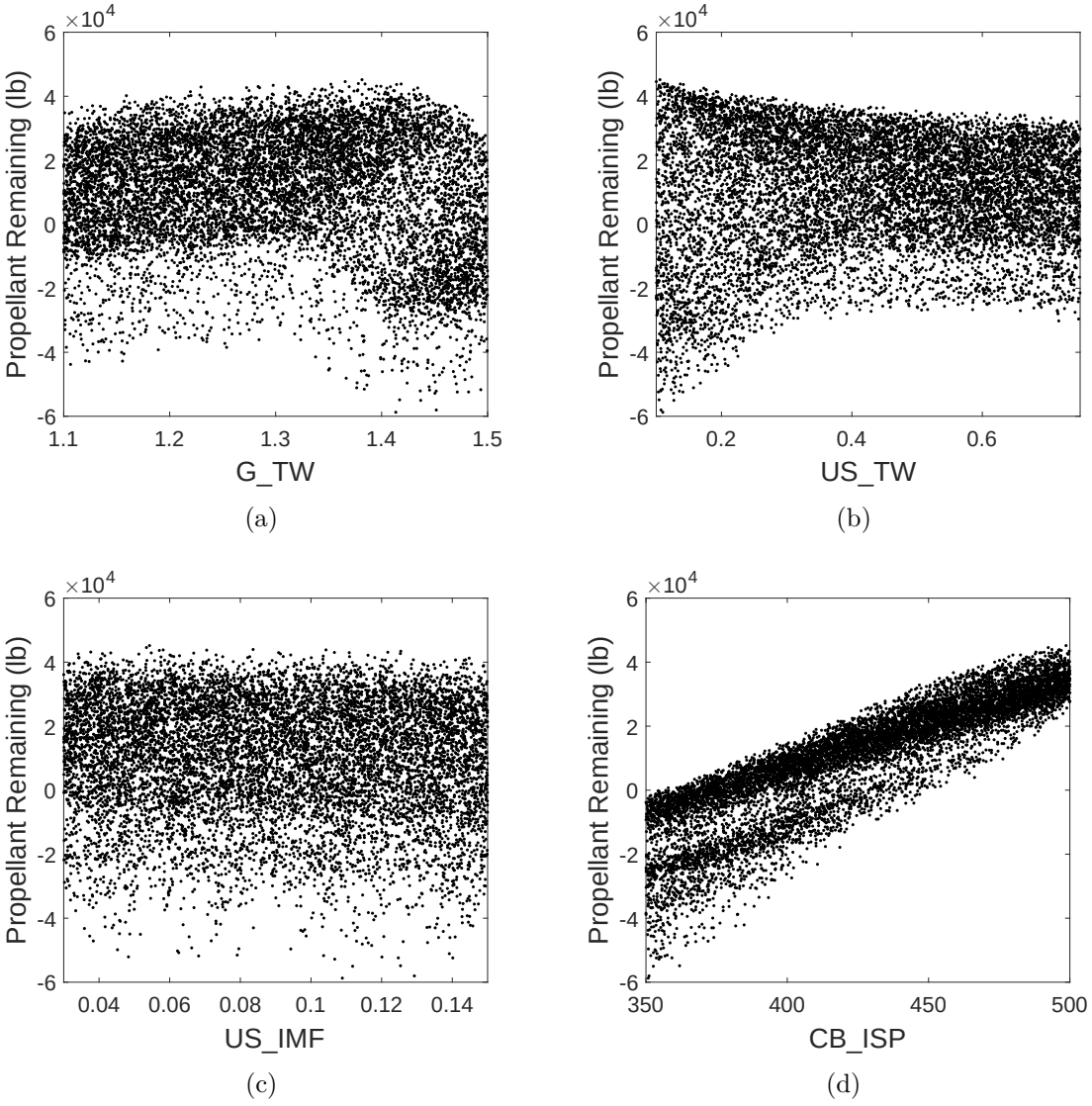
G_TW	US_TW	US_IMF	CB_ISP
1.37	0.1	0.03	500

throttle-able engine is available, it could increase the performance. Considering only a handful of vehicles, as is traditionally done in conceptual design, would not have revealed this unintuitive result and the simulation assumption that caused it. For now, however, it will be assumed that the engine cannot be throttled.

Another way to explore the design space is to perform a Monte Carlo analysis, which consists of randomly selecting points in the design space to analyze. Because evaluated a surrogate model is computationally inexpensive, thousands of points can be evaluated. Figure 88 shows the performance of $10,000$ points plotted against each of the design parameters. Monte Carlo analysis can be used to recognize trends in the design space or to find optimal designs. In this case, a strong trend is seen with respect to the ISP of the core and boosters.

The optimization of the propellant remaining is the final example of design space exploration is given here. The constrained optimizer in MATLAB, *fmincon* [109], can be used to find the best solution within the bounds. The optimized solution is given in Table 40. The values for CB_ISP and US_IMF are expected results, as they represent maximizing efficiency and minimizing weight respectively. The value for US_TW was discussed previously, and is therefore not unexpected. It is interesting that the optimized value for G_TW lies in the middle of the design variable range.

Using surrogate models in analysis can lead to a danger that will be mentioned here. The surrogate model only applies in regions where there is data. For example, no data exists for G_TW values greater than about 1.4 (see Figure 79). This means the surrogate model is extrapolating in certain regions, and cannot be relied upon. The designer must be careful to understand the limitations of a surrogate model and handle the results appropriately.



(a) (b) (c) (d)

Figure 88: Monte Carlo simulation for extended design study

4.3.2 Payload Contours

A useful way to visualize the trends captured in the performance model is to generate payload contour plots. The payload can be calculated directly from the propellant remaining. Propellant remaining represents unused propellant that can be replaced with payload. It is assumed that ullage and margin propellant is accounted for elsewhere. A baseline value for the payload of $63,500\ lb$ was given in Table 2, and the propellant remaining can simply be added to that. The payload in orbit will then depend on the design variable values. Contour plots for the different design variable combinations will be useful in understanding the design space. Figure 89 shows these payload contours for all the design variable combinations. The numbers on the contour lines represent the payload in orbit. Some of the plots are less interesting than others. For example, Figure 89(e) is relatively predictable. On the other hand, Figure 89(b) shows an island of best performing vehicles. It should be noted that these contours are two dimensional slices at specific values for the other variables. In this case, the values are set to the baseline Delta IV Heavy values, given in Table 5. A different point in the design space would lead to different contour plots. Another interesting trend is seen in Figure 89(d). The payload will either increase or decrease with US_TW depending on the value of CB_ISP. As discussed earlier, lower values of US_TW can be beneficial based on the assumptions used to model this vehicle. The benefits, however, depend on the value of CB_ISP. This is because CB_ISP will determine the staging point where the upper stage is ignited.

The plots included here provide a very good example of how the design space can be visualized for use in real time decision making. Conceptual designers can use these plots to identify areas of interest for further investigation.

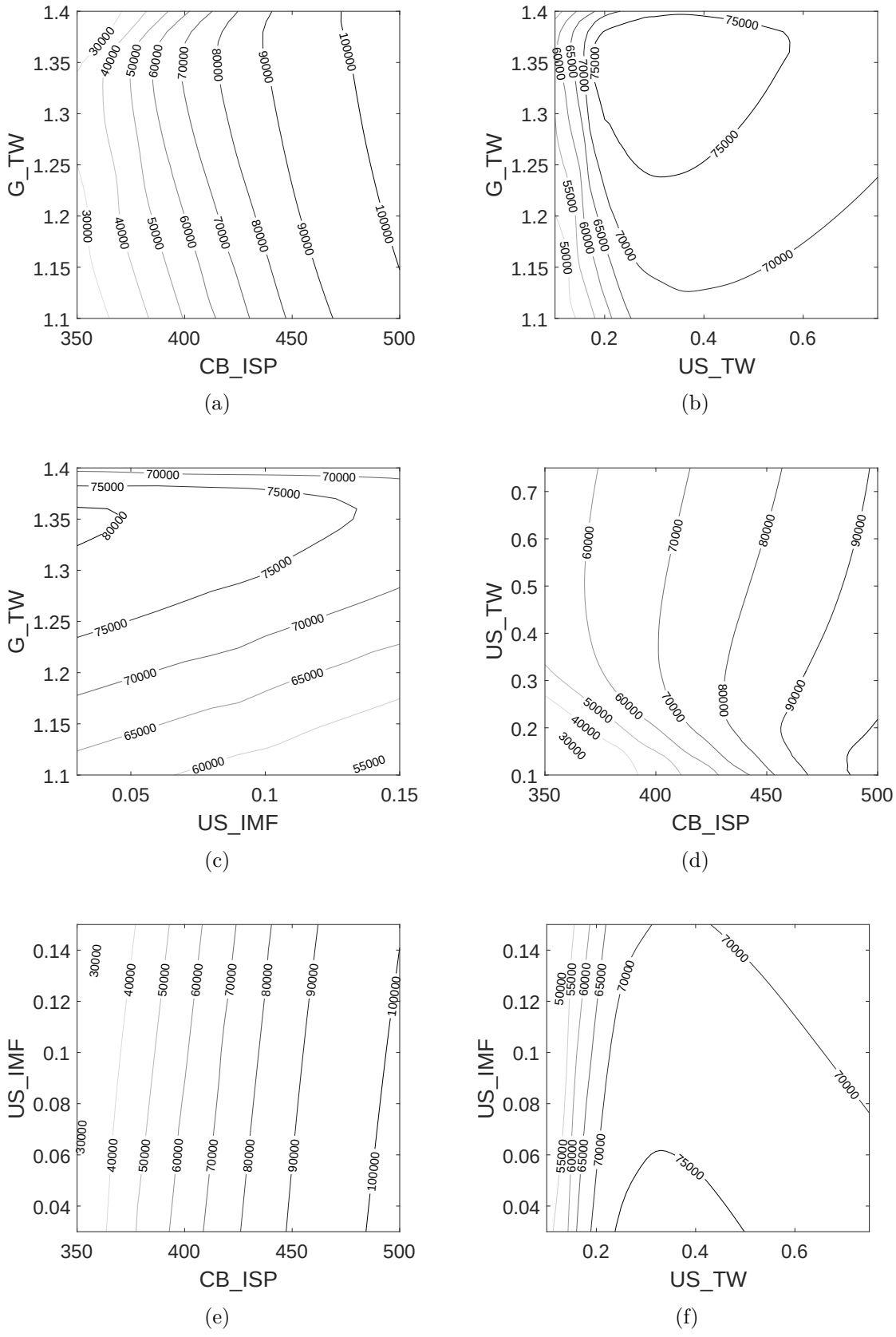


Figure 89: Payload contours for the extended design study

4.3.3 Other Applications

There are many other examples of potential uses for the performance model. Probabilistic analysis can be performed by generating distributions on the input variables. These distributions represent the likelihood of certain design variable values and can be used to randomly generate cases to evaluate, similar to the Monte Carlo analysis. In the Monte Carlo analysis, however, the distributions are uniform. In probabilistic analysis, the output will be represent the distribution of performance values based on the design variable distributions. This output distribution can be used to determine the likelihood of meeting a certain performance requirement. Design changes can be implemented or requirements modified if the likelihood is unacceptably low.

The performance model can also be used in conjunction with a cost model to evaluate what designs may be ideal from both a performance and cost perspective. If only the performance is considered, a very expensive design will likely result. Considering both performance and cost can help ensure the design is able to meet all the requirements while remaining in the specified budget.

A final application is mentioned regarding technologies. Technology infusion is used in many conceptual design studies as a way to close the gap between the performance and the requirements. The performance model generated in this thesis provides a way of evaluating what technologies will be most useful. In addition, a combination of technologies can easily be evaluated. It may not be possible to increase the CB_ISP value by a certain amount. However, increasing it by a small amount may change how another technology that operates on the US_TW affects the performance.

The utility of a performance model such as the one provided by the RAPTOR methodology is incredible in the context of conceptual design. It is the author's hope that the examples provided in this section will motivate any readers involved in launch vehicle conceptual design to carefully consider implementing and applying the RAPTOR methodology to aid in the design process.

CHAPTER V

CONCLUSIONS

Trajectory analysis in launch vehicle conceptual design provides the link between a specific vehicle concept and the capability of that vehicle concept. Currently, trajectory analysis is accomplished through a manual process that relies on a subject matter expert at various stages in the process. In the context of conceptual design, the current process restricts the number of vehicle concepts that can be evaluated. Ideally, the entire design space of interest would be considered. Instead, the number of vehicles is limited to a handful. The research objective of this thesis, repeated below, was designed to address this limitation.

Research Objective

Enable rapid and accurate launch vehicle performance evaluation in the context of conceptual design.

This research objective was introduced in Chapter 1, along with a road-map for how to meet the objective. The approach required the generation of a surrogate model, which relies on a set of data. Several research questions were developed in Chapters 2 and 3 to guide the process of meeting the research objective. The first question, stated in Section 2.4, addressed how to generate the data required for a surrogate model. Given the amount of literature on this topic, the question was answered without an experiment. The method selected to generate launch vehicle performance data is to employ a global initialization with a local direct optimization

method to find optimized trajectories, which is a common approach in the literature and in industry for this problem.

This method of trajectory optimization requires a control structure that determines how the control parameters are applied to the trajectory. Currently, this control structure is determine *a priori* by a trajectory analyst. While this may be feasible for the analysis of a single or even a few vehicles, this method is not readily applied when a large number of vehicles is considered. This led to a set of two sub-questions. The first addressed how to find the control structure for a trajectory optimization problem, and the second addressed how to apply it when many vehicles are considered. The OEPI method was developed to answer the first question. This method, which is based on a metric called the derivative difference, iteratively generates control structures by adding parameters in a way that most benefits the objective function given the assumptions. The OEPI method was successfully applied to three different problems, including a launch vehicle trajectory problem, thereby substantiating Hypothesis 2.1, restated below.

Hypothesis 2.1 - If a method based on the derivative difference is used to iteratively generate control structures, then that method will find control structures that maximize improvement at each iteration.

The application of the OEPI method to a design space of vehicles was the subject of the second sub-question. The goal was to find a control structure that was usable, defined as resulting in a change of less than 100 lb given the addition of a single parameter for any vehicle in the design space. For that, a relevant application problem was formulated using the Delta IV Heavy as a baseline vehicle. A design space defined by four parameters was identified in Section 2.5.2 as the design space of interest for

this problem and a set of vehicles was chosen to represent the design space in the experiments. Initially, it was expected that a usable control structure for the baseline vehicle would be usable for all vehicles, as stated in Hypothesis 2.2 below. However, this turned out to be incorrect. Instead, the OEPI method was applied simultaneously to the set of representative vehicles to result in a control structure that could be applied to every vehicle in the design space.

Hypothesis 2.2 - If the design variables being considered are continuous and represent reasonable vehicles, then a usable control structure for the baseline will be usable for other vehicles in the design space.

Once the control structure was determined, the optimization method could be applied. One of the challenges with the direct method for trajectory optimization is that the final result is dependent on the initial guess. The global initialization that is paired with the direct optimization mitigates this effect but not entirely. To address this challenge, the general optimization problem was posed as a statistical problem of finding the extreme values of a distribution representing the objective function values in the design space of interest. To this end, EVT was introduced in Section 3.2.4 with a corresponding research question regarding how to apply EVT to the optimization

problem. The resulting hypothesis is restated below.

Hypothesis 3 - If a random global search is performed multiple times to find the optimum of a function, then the resulting distribution of best results will be a GEV distribution.

Hypothesis 3 was tested by applying random searches to several test optimization functions and in every case but one, the resulting distribution was a GEV distribution. With this result, a random search could be applied to the trajectory optimization. However, a random search alone is ineffective in finding feasible trajectories, and therefore it is paired with the direct method for local optimization. The application of EVT to the trajectory optimization problem when a random search was combined with the direct method was the subject of the fourth research question. This was subdivided into two sub-questions. The first addressed if EVT could be applied to the trajectory optimization problem. The corresponding hypothesis is restated below.

Hypothesis 4.1 - If the underlying performance population for a given vehicle is used to generate a distribution of sample maxima, that distribution will be distributed as a GEV distribution and therefore the population will be in the domain of attraction of the GEV distribution.

Hypothesis 4.1 was tested by finding the underlying performance population for each vehicle in the set of representative vehicles used in this thesis. In every case, the distribution of sample maxima resulted in a GEV distribution. This means that the

best results of repeated random searches for the launch vehicle trajectory optimization problem would result in a GEV distribution. However, this approach would be very inefficient. Consequently, direct method optimization is employed in conjunction with a random search. It was hypothesized that the results of the local optimization process can be viewed as analogous to the results of the maximum of a random search. This was the subject of the second sub-question and the corresponding hypothesis, restated below.

Hypothesis 4.2 - If a random search is used to initialize repeated direct optimization processes for a trajectory problem, then the results will be analogous to a distribution of sample maxima and distributed as a GEV distribution.

After repeatedly applying the direct optimization process to each of the representative vehicles, it was concluded that the optimization results were not exactly analogous to the sample maxima. However, after quantifying the error in several ways, it was determined that the distribution of optimized results is a very close estimate of the distribution of sample maxima. For one of the error estimates, the difference was less than 10 lb when estimating the vehicles performance, which was on the order of $10,000$ lb.

The reason the distribution of optimized results is useful is to estimate the performance of a vehicle. Because this is a statistical method, the possibility of inaccurate estimates exists. Therefore, a statistical method called bootstrapping was employed to quantify the uncertainty in a given performance estimate. The results showed that

the bootstrap method can consistently quantify the uncertainty, thereby substantiating Hypothesis 5, repeated below.

Hypothesis 5 - If bootstrapping is used to measure uncertainty in the performance metric, then larger error in the performance metric will be indicated by higher uncertainty.

The successful completion of the experiments described resulted in a methodology for evaluating the performance of a vehicle in the design space being considered. The research goal, however, was to rapidly enable this evaluation, which can be achieved by generating a surrogate model. A set of vehicles to evaluate is required for the generation of a surrogate model. The overall methodology presented in this thesis results when all these elements are combined, as seen in Figure 73 in Section 3.6. The question of how to select the vehicles to evaluate for the surrogate model was answered from the literature. The field of DOE was applied to generate LHC designs. The only remaining step was to generate the surrogate model. Several options were considered. Given the nature of the problem, it was hypothesized that neural networks would provide the best fits. This was stated in Hypothesis 7, repeated below. After fitting all of the surrogate models listed, neural networks did indeed provide the best

estimates of the data.

Hypothesis 7 - If a neural network is used to create a surrogate model for the performance data, the neural network will provide a better fit to the data than polynomial regression, kriging models, radial basis functions, or support vector regressions.

The research objective of this thesis was to enable rapid and accurate launch vehicle performance evaluation in the context of conceptual design. This was achieved by implementing the process shown in Figure 73. The experiments discussed in Chapters 2 and 3 and reviewed above tested each of the elements of the overall process as it was implemented. In practice, an accurate surrogate model fit for the design space considered in this thesis is achievable in around 40 hours on a Dell OptiPlex 790 with a Core i7 processor. The process does not require a human in the loop. The analyses are easily parallelized and a small computer cluster with as few as four machines like the one used can bring the runtime down to less than 10 hours. The surrogate model created enables rapid and accurate performance analysis for launch vehicles in the context of conceptual design. It is rapid because it is a closed form equation that can be evaluated by a computer in fractions of a second. It is accurate because it represents the performance resulting from an industry standard trajectory optimization tool that takes into account trajectory losses.

The overall methodology is named Rapid Trajectory Optimization Routine, or RAPTOR. RAPTOR is implemented in an extended design study in Chapter 4. This extended design study is used to provide an example implementation, compare the process to current methods, and showcase some possible applications. When compared to the current methods of evaluating performance metrics for a launch vehicle, the RAPTOR method is the only one that successfully provides an accurate

picture of the design space in a short amount of time. Traditionally, a trajectory expert manually sets up the analyses and inspects the results. The control structure is determined *a priori* and modified manually if deemed necessary. The initial guess for the analysis is manually based on previous trajectories and/or subject matter expertise to find the best performance. The result of this process, which requires anywhere from one to several hours of a trajectory expert's time, is the performance value for a single vehicle. The method presented in this thesis provides a way of estimating the performance for thousands of vehicles in the design space considered virtually instantaneously. This affords decision makers the ability to weigh options and ask the "what if" questions in a real time setting with accurate estimates of the performance of the launch vehicle considered.

5.1 Summary of Contributions

The work described in this document represents several contributions to the field of launch vehicle performance analysis during conceptual design. The first is the method developed to generate control structures for a trajectory optimization problem addressing Research Question 2.1. The method was shown to be effective on a simple optimal control problem, a lunar launch problem with assumptions, and an ETO launch vehicle trajectory problem. As far as the author knows, this is the first repeatable and traceable method for generating control structures for direct method optimization problems. The method and results are being submitted as "Iterative Generation of Parametric Control Structures for Trajectory Optimization." This submission will be referred to as [S]. This method was implemented for the design space considered in this thesis to find a control structure that could be used for all the vehicles to address Research Question 2.2.

The second contribution is the application of EVT to optimization, addressed

by Research Question 3, and specifically to the trajectory optimization problem, addressed by Research Questions 4.1 and 4.2. A previous study has suggested the applicability of EVT to optimization, but the tests performed were minimal and assumed knowledge of the optimum. In this study, EVT is applied to a number of varied test functions. Results show that EVT is indeed applicable by using random searches. A further step is taken in applying EVT to the trajectory optimization problem, where optimization is used instead of a random search. The results show that while some of the assumptions of EVT are violated, it still provides an excellent approximation of the resulting data. The application of EVT to the trajectory optimization problem has been published by the author in "Launch Vehicle Performance Analysis using Extreme Value Theory" in the AIAA SPACE 2015 conference [161].

The final and most important contribution is that of the overall method that results in generating a surrogate model for the launch vehicle performance, shown in Figure 73. Generating a surrogate model is not a new process. However, generating accurate data for launch vehicle performance in an automated fashion is a contribution. In addition, several different surrogate model types were compared to address Research Question 7. This has tremendous potential in conceptual design. Technologies can be compared against each other quantitatively. Sensitivities of parameters, requirements, assumptions can be mathematically measured. Probabilistic analysis and Monte Carlo simulations are feasible. This can help greatly as managers are continuously asking "what if" questions. A surrogate model enables a real time answer in conceptual design. A simplified implementation of some of the elements of this method was published by the author in "A Method for Launch Vehicle Performance Analysis via Surrogate Modeling" in 2016 in the AIAA Modeling and Simulation Technologies Conference [160]. This publication compared several different metrics, some of which resulted from using EVT in the process, for generating a surrogate model. The conclusion was that the 95th percentile from the GEV distribution resulted in

the best fit.

Table 41 maps the contributions in this dissertation to the author's relevant publications as well as the corresponding Research Question and the section of this document in which it is addressed. Some of the relevant publications are mapped to the research areas as well. It is notable that two of the columns to not have existing literature associated with them. This is because these contributions directly result from the contributions of this thesis. There is little literature about the application of EVT to optimization, and the author found no literature regarding the application of EVT specifically to trajectory optimization. The performance metric for a surrogate model resulted directly from the application of EVT to the trajectory problem, and consequently no existing literature was found.

5.2 Recommendations Future Work

There are several avenues of future research that can expand upon the work presented in this thesis. The overall method developed and implemented was applied to a design space made up of continuous variables. In launch vehicle conceptual design there exist some discrete variables that would be useful to include, such as number of boosters, number of stages, or number of engines. A process for including discrete variables would provide a useful capability. Another area of research could expand upon the OEPI method developed in Section 2.6.1. The OEPI method is limited to considering equivalent parameter substitutions. That limitation can be removed to indicate not only which parameter to split, but also exactly how to split it. Additionally, the control structure could be based on another variable, such as mass, instead of time. A third area would be to consider the optimization methods implemented in the tool used in this thesis. This method is not limited to a specific optimization algorithm, and faster more accurate optimization algorithms would decrease the time needed to implement the method in this thesis. The fourth area listed here is to implement

Table 41: Summary of contributions

Reference	Control Structures		EVT		Performance Surrogate	
	OEPI Method	Application to Design Space	Optimization	Trajectory Optimization	Performance Metric	Model Selection
Betts [19]	✓					
Hull [78]	✓					
Waters [176]	✓	✓				
Beirlant [16]			✓			
Hüsler [80]			✓			
Qazi [129]						✓
Akhtar [2]						✓
Steffens [S]	✓					
Steffens [161]				✓		
Steffens [160]					✓	
Research Question	2.1	2.2	3	4.1 and 4.2		7
Dissertation Section	2.6.1	2.6.3	3.3	3.4	3.4	3.7

adaptive sampling methods as a way of exploring the design space. Other DOE methods could be applied, both for the vehicles and the initial guesses. A general method for selecting good initial guesses for optimization problems would prove very useful. The final area of future research suggested here is to implement this method on new launch vehicle designs. This method enables, among other things, probabilistic analysis, technology trade studies, and rapid multidisciplinary analysis. Applying this method to new vehicle designs can result in faster turnaround times for conceptual design studies and yield insight into the launch vehicle performance design space.

APPENDIX A

SAMPLE DISTRIBUTION DATA

Table 42 shows the results of repeated optimization solutions to the baseline vehicle described in Section 2.5. The results were used as represented data for some of the methodology development in Chapter 3. The data is simply the remaining propellant in orbit in pounds for each of the successful repetitions.

Table 42: Propellant remaining (lb) for repeated optimization runs for the Delta IV Heavy

766.5	856.4	842.5	701.3	775.5	786.3	840.4	855.7	724.7	864.5
813.4	678.7	854.6	778.2	694.7	813.6	808.7	824.8	779.4	832.1
675.5	824.7	788	839.6	681.1	715.4	866	718.4	693.9	852.6
614.8	805	806	843.9	795.9	729.2	704.6	801	826.1	845
795.5	771.3	792.6	535	841.4	375.4	832.3	762.7	836.3	665.1
716.2	774.5	625.9	870.6	817.8	856.1	814.5	850.6	794.2	766.2
670.4	682.4	797.6	875.5	680.2	787.8	846.5	841.9	855.8	721.7
836.5	487.2	722.1	685.8	585.2	699.2	755.6	762.3	687	705.4
845	795.5	542.3	775.3	581	730.4	777.2	820.8	734	696.5
829.9	706.3	824.9	798.8	742.5	197.5	756.7	779.4	829.5	719.7
783.7	722.2	803.4	727.5	858	784.4	693	819.5	862.5	759.1
831.8	689.9	737.9	692.7	675.7	661.9	771.7	501.7	817	724.5
796.2	714.3	693.7	751.3	669.9	653.3	794.2	836.5	779.8	857.1
803.7	834.4	854.8	768.3	845.8	805.8	766.3	801.6	816.6	781.8
829.3	897.3	832.2	841.2	793	751.9	742.4	808.4	859.4	799.1
834.8	675.3	741.1	738.9	803.8	645.4	811.6	689.7	733.6	732.4
778.3	846.9	868	636.4	803.2	741.4	704.5	856	726.3	780
806	782.5	813	803.5	847.8	658.2	840.2	823	675.4	841.3
864.5	771.5	783	831.9	826.1	891.2	676.3	795.9	712.9	828.9
843.6	798.2	713	589.2	657	781.1	863.4	805.6	651.7	766.5
371.4	779.6	858.9	617.7	836.2	770	830.8	889.9	869.3	813.5
796.1	753.1	825.8	697.7	794.6	849.3	814	847.8	693.2	828.4
835.9	810.1	833.7	610.8	866.9	682.8	657.8	822.1	820.6	808.7
837	796.3	704.6	796.9	576.3	575.9	677.6	875.7	703.7	730.6

808.8	746.9	851.9	738.1	653.9	849.9	700.8	782.4	717.6	819.6
849.9	819	660.5	806.8	646.5	847.1	758.8	645.5	843.6	813.2
843.1	648.6	765.4	678.1	847.9	829.2	818.7	787.1	831.1	860
866.3	761.4	713.4	641.7	826.1	844.1	819.3	778.4	688.5	778.4
811.6	629.2	697.3	646.1	820	853.5	796.7	693.2	809.9	755.3
830.3	682.2	728.8	852.5	792.5	859.9	693.7	816.5	826.8	741
726	812.9	746.8	659.9	832.6	588.9	743.1	847.8	821.9	852.3
609.3	602.8	825.2	757.8	851.2	838	799.5	778.2	770.6	873.7
799	729.6	836.1	832.1	620.8	688.8	760.7	779.1	682.1	703.2
818.9	667.6	827.1	883.5	815.4	799.9	689.9	813.5	772.9	863.6
833.8	777.7	861.7	785	768.1	860	849	833.2	664.1	823.4
845.9	754.2	804.9	665.6	531.3	792.2	800.6	849.7	773.7	694.9
803.1	824.8	832.2	758.6	828.3	847.8	726.5	731.9	796.3	784
687.1	849.9	786.2	733	621.3	826.6	841.4	774.8	800.4	793.4
712.8	515.4	833.5	722.6	621.1	858.2	674.4	839.3	779.6	835.1
788.2	831.8	416.2	814.6	812.4	783.7	864.7	735.3	594	812.9
746	702.8	831.9	763.1	853.4	855.6	783.3	619	667.4	826.1
789.1	631.3	841.5	784.7	752.2	818.6	828.1	864.2	869.3	822.7
765.6	855.3	698	762.3	831.4	840.4	795	880.4	885.3	712.3
799.9	776.6	872.4	856.1	559.2	747.5	699.2	684	810	829.2
803.1	879.1	820.8	766.5	804.1	803.4	788.7	860.1	854.6	711.5
774.6	818.4	804.9	752.4	777.5	669.3	847.5	833.3	661.9	396.9
772.9	714.5	746.6	695.2	824.9	828.3	795.4	843.9	824.5	856.8
698.8	741.8	655.4	808.3	907.5	593.9	859.5	826.6	564	817.2
794.4	871.8	822	727.7	826.7	844	791	704.8	735.9	701.7
770	804.1	649.9	879.3	825.2	659.3	751.2	549.6	676.7	785.2
857.8	832.2	815.2	706.8	661.5	821.3	664.2	842.5	701	708.7
811.5	781.6	807.6	723.8	811.2	497.7	784.5	875.6	611.4	850.4
760.3	772.4	759.8	702.4	875.4	753	736.1	785	827.5	781.7

APPENDIX B

DELTA IV HEAVY BASELINE TRAJECTORY INPUT

The POST input file for the baseline Delta IV Heavy vehicle and mission is included here. This input file is used to generate the trajectory in Section 2.5.1.

```
$SEARCH
srchm    = 4,
optvar   = 'WEIGHT',
opt      = 1,
optph    = 1000,
maxitr   = 100,
coneps   = 91,
ioflag   = 0,
wopt     = 0.00001,
nindv    = 8,
indvr    = 6HPITPC2, 6HPITPC2, 6HPITPC2, 6HPITPC2,
           6HPITPC2, 6HPITPC2, 6HPITPC2, 3HAZL,
indph    = 5,20,35,52,60,70,80,1,
u        =  -0.148,0.06,-0.06,-0.07,-0.05,-0.04,-0.04,86.2,
ndepv    = 4,
depvr    = 5HGCRAD, 6HGAMMAI, 3HINC, 5HXMAX1,
depval   = 22237862,0,28.7,500,
depph    = 100,100,100,100,
idepvr   = 0,0,0,1,
deptl    = 100,0.01,0.01,1,
$
$GENDAT
event    = 1,
maxtim   = 3400,
altmax   = 1e12,
altmin   = -1000,
prnc     = 0,
prnca    = 0,
fesn     = 1000,
dt       = 1,
pinc     = 1,
time     = 0,
nstpl    = 1,
nstph    = 5,
istepf   = 1,1,1,1,1,
wstpd(1) = 118000, / Boosters
wstpd(2) = 59000, / Core
wstpd(3) = 7700, / Upper Stage
wstpd(4) = 63500, / Payload
wstpd(5) = 6470,  / Fairing
wprp(1)  = 880000, / Booster prop
wprp(2)  = 440000, / Core prop
wprp(3)  = 60000, / Upper stage prop
```

```
44    menstp   = 1,2,3,
45    mentnk   = 1,2,3,
46    sref = 219,
47    neng     = 3,
48    nengh    = 3,
49    nengl    = 1,
50    iengmf   = 1,1,0
51    ienga    = 0,1,0,
52    iwdf     = 3,3,3,
53    iwpf     = 1,1,1,
54    iguid(1) = 1,
55    iguid(2) = 0,
56    iguid(4) = 1,
57    pitpc(1) = 0,
58    pitpc(2) = 0,
59    npc(1)   = 2,
60    npc(5)   = 5,
61    npc(7)   = 1,
62    npc(9)   = 1,
63    npc(8)   = 2,
64    npc(30)  = 3,
65    gclat    = 28.46,
66    long     = 279.38,
67    gdalt    = 0,
68    velr     = 0,
69    azvelr   = 0,
70    azl      = 84,
71    gammar   = 0,
72    piti     = 0,
73    roli     = 0,
74    yawi     = 0,
75    monx(1)  = 'dynp',
76    asmax    = 5,
77    $
78    $TBLMLT
79    cdm(1) = 1,
80    $
81    $TAB
82    table='genv3t',1,'dynp',2,3*1, / genv3 = dynp x k-factor
83    0, 0,
84    1000, 2000,
85    $
86    $tab
87    table='genv4t',1,'vela',2,3*1, / genv4 = vela x units
88    0, 0,
89    100000, 128.593,
90    $
91    $tab     / genv5 = dynp x vela x k-factor x units
92    table='genv5t',2,'genv3','genv4',2,2,8*1,
93    0, 0, 0,
94    128.593, 0,
95    2000, 0, 0,
96    128.593,257186,
97    $
98    $tab
99    table = 'genv1t',0,1,
100   $
101   $tab
```

```
102     table = 'genv2t',0,1,
103     $
104    *include 'aero_delta_iv_heavy_w_boosters.aero'
105    *include 'prop_delta_iv_heavy.prop'
106     $GENDAT
107     event    = 5,
108     critr    = 'gdalt',
109     value    = 2000,
110     iguid(4) = 0,
111     dtimr(3) = 1,
112     timrf(3) = 0,
113     endphs   = 1,
114     $
115     $GENDAT
116     event    = 10,
117     critr    = 'dynp',
118     value    = 150,
119     iguid(1) = 0,
120     iguid(2) = 1,
121     iguid(6) = 3,
122     iguid(7) = 0,
123     iguid(8) = 0,
124     desn     = 15,
125     dalpha   = 0,
126     betpc(2) = 0,
127     bnkpc(2) = 0,
128     dtimr(1) = 1,
129     timrf(1) = 0,
130     endphs   = 1,
131     $
132     $GENDAT
133     event    = 12,1,
134     critr    = 'gdalt',
135     value    = 12000,
136     dtimr(2) = 1,
137     timrf(2) = 0,
138     $
139     $tblmlt
140     $
141     $tab
142     table = 'genv2t',1,'timrf2',2,3*1,
143     0, 1,
144     5, 0.545,
145     endphs = 1,
146     $
147     $GENDAT
148     event    = 13,1,
149     critr    = 'timrf2',
150     value    = 5,
151     $
152     $tblmlt
153     $
154     $tab
155     table='genv2t',0,0.545,
156     endphs=1,
157     $
158     $GENDAT
159     event    = 15,
```

```
160     critr     = 'timrf1',
161     value     = 10,
162     iguid(1) = 0,
163     iguid(2) = 0,
164     iguid(4) = 0,
165     alppc(2) = 0,
166     betpc(2) = 0,
167     bnkpc(2) = 0,
168     endphs    = 1,
169     $
170     $GENDAT
171     event     = 20,
172     critr     = 'dynp',
173     value     = 20,
174     mdl       = 3,
175     iguid(1) = 1,
176     iguid(2) = 0,
177     iguid(4) = 0,
178     endphs    = 1,
179     $
180     $GENDAT
181     event     = 25,
182     critr     = 'wprp1',
183     value     = 15000,
184     mdl       = 3,
185     $
186     $tblmlt
187     $
188     $tab
189     table = 'genv1t',1,'wprp1',2,3*1,
190     15000, 1,
191     4002.885085574573, 0.545,
192     endphs = 1,
193     $
194     $GENDAT
195     event     = 26,1,
196     critr     = 'wprp1',
197     value     = 4002.885085574573,
198     mdl       = 3,
199     $
200     $tblmlt
201     $
202     $tab
203     table='genv1t',0,0.545,
204     endphs=1,
205     $
206     $GENDAT
207     event     = 28,
208     critr     = 'wprp1',
209     value     = 0,
210     mdl       = 1,
211     iengmf    = 0,1,0,
212     dtimr(1) = 1,
213     timrf(1) = 0,
214     endphs    = 1,
215     $
216     $GENDAT
217     event     = 29,
```

```
218     critr    = 'timrf1',
219     value    = 3,
220     mdl      = 1,
221     istepf   = 0,1,1,1,1,
222     $
223     $tblmlt
224     $
225    *include 'aero_delta_iv_med.aero'
226     $tab
227     endphs = 1,
228     $
229     $GENDAT
230     event    = 30,
231     critr    = 'timrf1',
232     value    = 4,
233     mdl      = 1,
234     dtimr(1) = 1,
235     timrf(1) = 0,
236     $
237     $tblmlt
238     $
239     $tab
240     table = 'genv2t',1,'timrf1',2,3*1,
241     0, 0.545,
242     5, 1,
243     endphs = 1,
244     $
245     $GENDAT
246     event    = 40,1,
247     mdl      = 9,
248     critr    = 'genv5',
249     value    = 0.1,
250     nstph    = 4,
251     endphs   = 1,
252     $
253     $GENDAT
254     event    = 35,
255     critr    = 'timrf1',
256     value    = 5,
257     mdl      = 1,
258     $
259     $tblmlt
260     $
261     $tab
262     table = 'genv2t',0,1,
263     endphs   = 1,
264     $
265     $GENDAT
266     event    = 50,
267     mdl      = 1,
268     critr    = 'wprp2'
269     value    = 0,
270     iengmf   = 0,0,0,
271     ienga    = 0,0,0,
272     dtimr(1) = 1,
273     timrf(1) = 0,
274     endphs   = 1,
275     $
```

```
 $GENDAT
 event    = 51,
 critr    = 'timrf1'
 value    = 5,
 nstpl    = 3,
 endphs   = 1,
 $
 $GENDAT
 event    = 52,
 critr    = 'timrf1'
 value    = 8,
 iengmf   = 0,0,1,
 ienga    = 0,0,1,
 dtimr(1) = 1,
 timrf(1) = 0,
 endphs   = 1,
 $
 $GENDAT
 event    = 60,
 mdl      = 1,
 critr    = 'timrf1'
 value    = 100,
 endphs   = 1,
 $
 $GENDAT
 event    = 70,
 critr    = 'timrf1'
 value    = 250,
 endphs   = 1,
 $
 $GENDAT
 event    = 80,
 critr    = 'timrf1'
 value    = 500,
 endphs   = 1,
 $
 $GENDAT
 event(1) = 100,
 critr    = 'veli',
 value    = 25159.4798,
 endphs   = 1,
 $
 $GENDAT
 event        = 1000,
 critr        = 'tdurp',
 value        = 0,
 endphs       = 1,
 endprb       = 1,
 endjob       = 1,
 $
```

APPENDIX C

TRAJECTORY DESIGN SPACE TRANSFORMATIONS

In Section 2.5.2 two separate parameterizations were presented. This appendix provides the details for the transformation between the second parameterization and the first. Table 43 gives the variables for the two parameterizations in no particular order. The design variables labeled $8 - 15$ are the same in both parameterizations, so not transformation is needed.

The Upper Stage Prop can be calculated using Equation 96.

$$\text{Upper Stage Prop} = \frac{1 - \text{Upper Stage IMF}}{\text{Upper Stage IMF}} \times \text{Upper Stage Burnout} \tag{96}$$

Now that the upper stage total mass is known, the Upper Stage Thrust can be calculated using Equation 97.

$$\begin{aligned} \text{Upper Stage Thrust} = \text{Upper Stage TW} \times \\ (\text{Upper Stage Prop} + \text{Upper Stage Burnout} + \text{Payload}) \end{aligned} \tag{97}$$

The Core Propellant is calculated using Equation 98.

$$\text{Core Prop} = \frac{\text{Core PMF}}{1 - \text{Core PMF}} \times \text{Core Burnout} \tag{98}$$

With the Core Prop known, the Booster Prop is then given by Equation 99.

$$\text{Booster Burnout} = \text{Core Prop} \times \text{Booster Core Propellant Ratio} \tag{99}$$

With the Booster Prop known, the Booster Burnout is calculated using Equation 100.

Table 43: Two sets of parameters for the Delta IV Heavy design space

	Parameterization 1
1	Vehicle TW
2	Upper Stage TW
3	Upper Stage IMF
4	Booster PMF
5	Core PMF
6	Booster Core Thrust Ratio
7	Booster Core Propellant Ratio
8	Upper Stage Burnout
9	Core Burnout
10	Core ISP
11	Booster ISP
12	Upper Stage ISP
13	Payload
14	Fairing
15	Maximum Dynamic Pressure

	Parameterization 2
1	Core Thrust
2	Upper Stage Thrust
3	Booster Burnout
4	Booster Prop
5	Upper Stage Prop
6	Booster Thrust
7	Core Prop
8	Upper Stage Burnout
9	Core Burnout
10	Core ISP
11	Booster ISP
12	Upper Stage ISP
13	Payload
14	Fairing
15	Max Dynamic Pressure

$$\text{Booster Burnout} = \frac{1 - \text{Booster PMF}}{\text{Booster PMF}} \times \text{Booster Burnout} \tag{100}$$

The final two variables are calculated using the total thrust and total weight, give in Equations 101 and 102.

$$\begin{aligned} \text{Total Weight} = &\text{Booster Burnout} + \text{Booster Prop} + \\ &\text{Core Burnout} + \text{Core Prop} + \\ &\text{Upper Stage Burnout} + \text{Upper Stage Prop} + \\ &\text{Payload} + \text{Fairing} \end{aligned} \tag{101}$$

$$\text{Total Thrust} = \text{Vehicle TW} \times \text{Total Weight} + \text{SL Correction} \tag{102}$$

The SL Correction term in Equation 102 is due to the reduction in thrust at sea level due to atmospheric pressure. Table 3 shoes the exit area for each of the 3 engines as 49.9 ft^2. The sea level correction term can then be calculated as 49.9

$ft^2 \times 3 \times 2116.24\ lb/ft^2$, where $2116.24\ lb/ft^2$ is sea level pressure. The value is $316927.9\ lb$.

Finally, the Core Thrust and Booster Thrust can be calculated as shown in Equations 103 and 104.

$$\text{Core Thrust} = \frac{\text{Total Thrust}}{1 + \text{Booster Core Thrust Ratio}} \tag{103}$$

$$\text{Booster Thrust} = \frac{\text{Total Thrust} \times \text{Booster Core Thrust Ratio}}{1 + \text{Booster Core Thrust Ratio}} \tag{104}$$

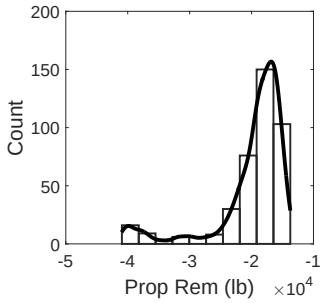
For this parameterization, the booster values were treated as the totals for both boosters. In other words, Booster Thrust is the total thrust from both boosters, or Booster Burnout is the burnout mass of both boosters.

APPENDIX D

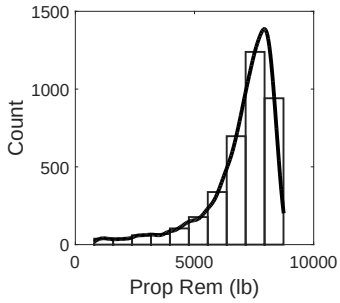
ADDITIONAL FIGURES

D.1 Nonparametric Fits for the Vehicle Performance Populations

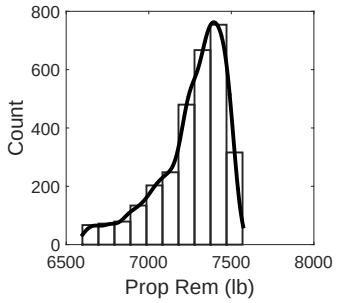
This appendix section includes the nonparametric fits for the vehicle performance populations used in Section 3.4.3.3. The lower outliers are excluded in these figures (see Section 3.4.2 for explanation of lower outliers). The nonparametric fits are generated using Kernel Density Estimates (KDE). The KDEs are sampled repeatedly to find the distribution of sample maxima for each vehicle and determine if that distribution is a GEV distribution. This process is explained in depth in Section 3.4.



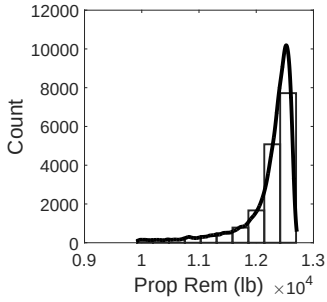
(a) Vehicle 1



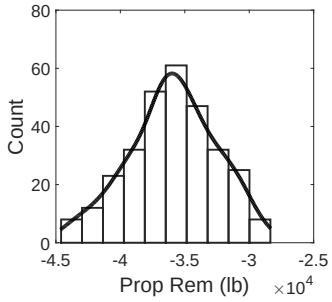
(b) Vehicle 2



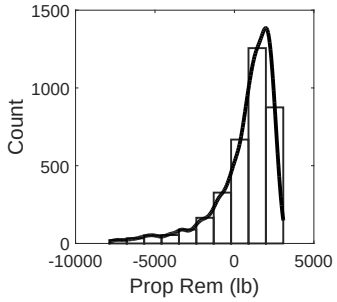
(c) Vehicle 3



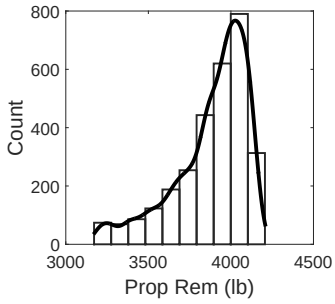
(d) Vehicle 4



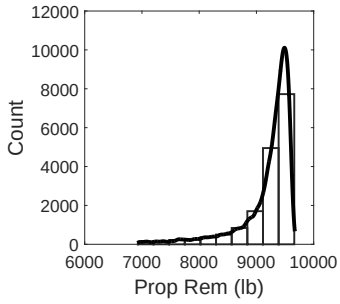
(e) Vehicle 5



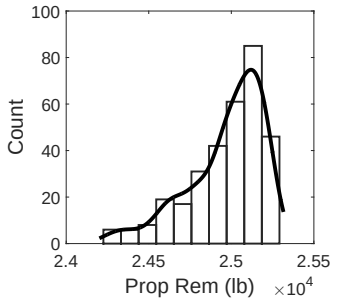
(f) Vehicle 6



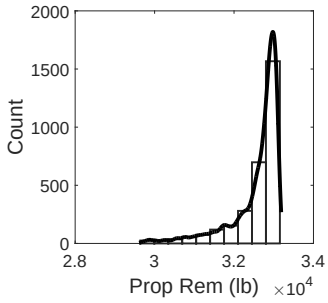
(g) Vehicle 7



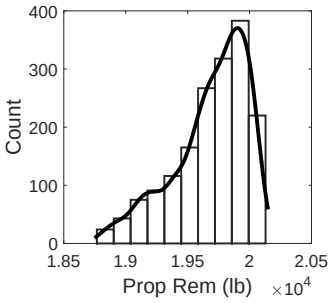
(h) Vehicle 8



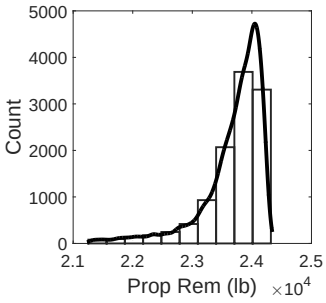
(i) Vehicle 9



(j) Vehicle 10

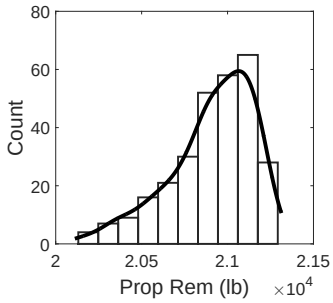


(k) Vehicle 11

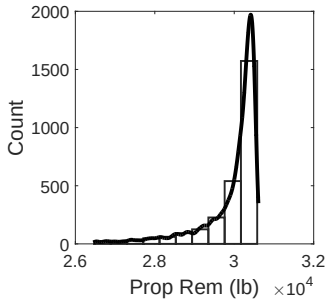


(l) Vehicle 12

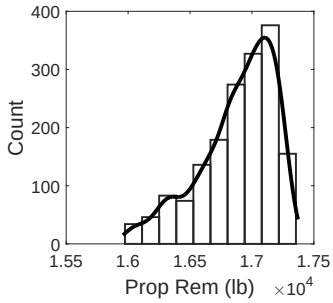
Figure 90: Performance populations with nonparametric fits for vehicles 1 - 12



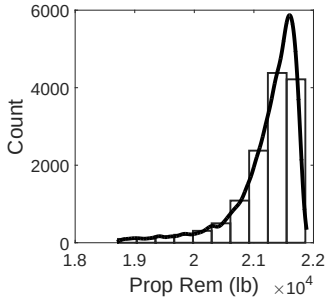
(a) Vehicle 13



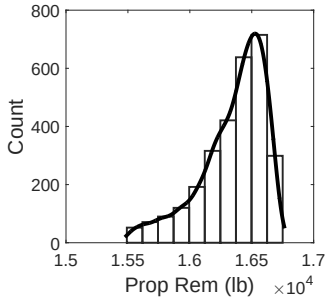
(b) Vehicle 14



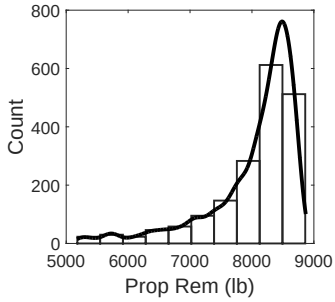
(c) Vehicle 15



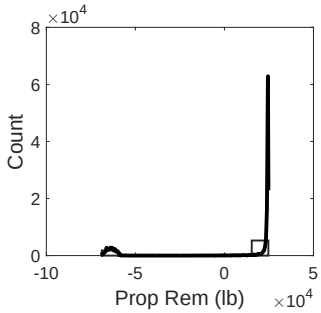
(d) Vehicle 16



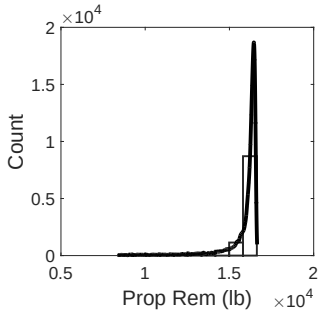
(e) Vehicle 17



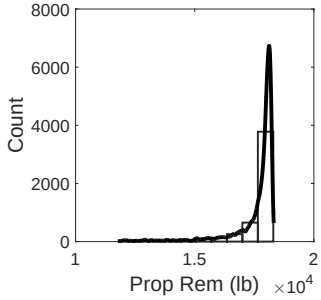
(f) Vehicle 18



(g) Vehicle 19



(h) Vehicle 20

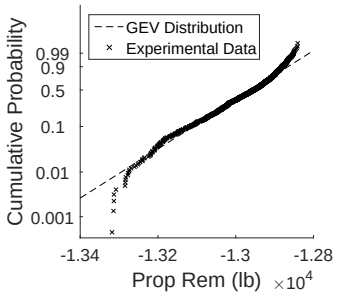


(i) Vehicle 21

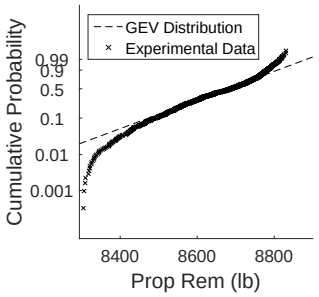
Figure 91: Performance populations with nonparametric fits for vehicles 13 - 21

D.2 Probability Plots for Repeated Optimization Analyses

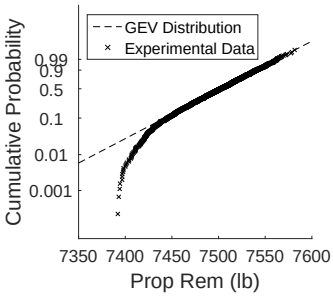
This appendix section shows the probability plots for the GEV fits and the repeated optimization analyses, where Prop Rem is the propellant remaining. These figures are similar to Figures 62 and 63. Instead of having quantiles on the y-axis, however, the cumulative probability, or percentiles, are shown. The phenomena seen with the lower regions of the distributions of optimized analyses explained in Section 3.4.7. The GEV distributions are predicting lower values than are seen for the lower percentiles.



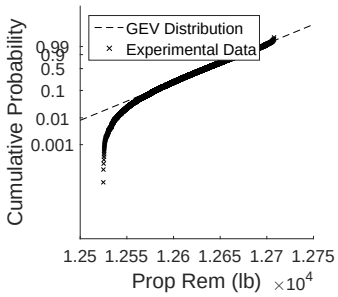
(a) Vehicle 1



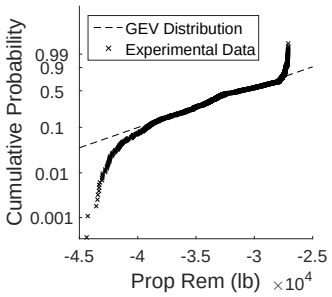
(b) Vehicle 2



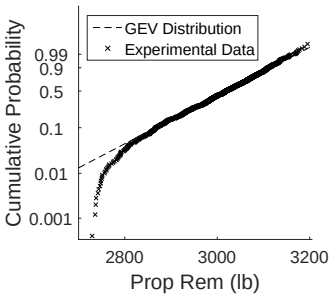
(c) Vehicle 3



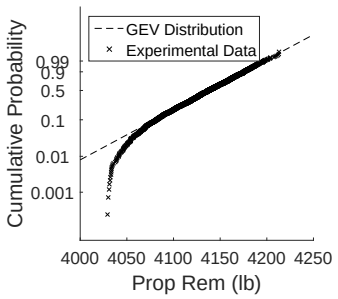
(d) Vehicle 4



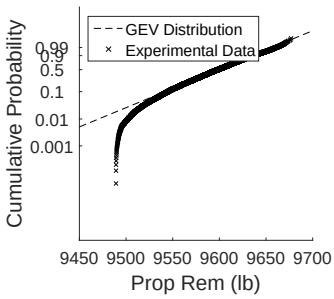
(e) Vehicle 5



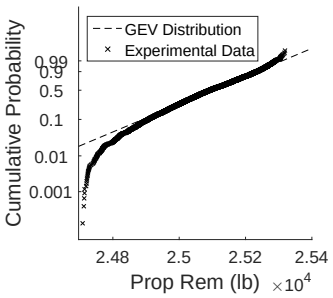
(f) Vehicle 6



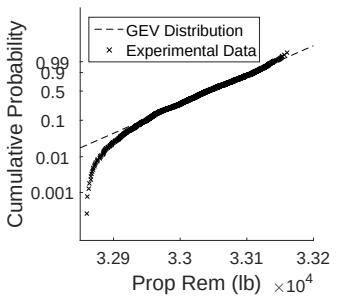
(g) Vehicle 7



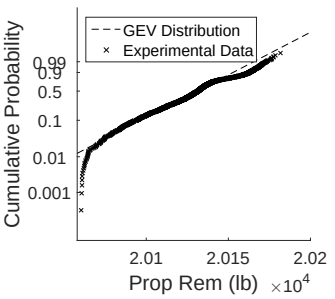
(h) Vehicle 8



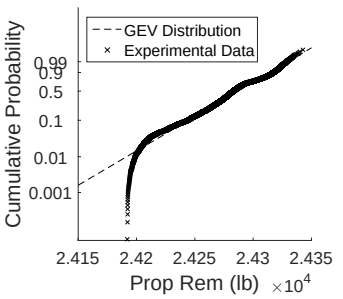
(i) Vehicle 9



(j) Vehicle 10

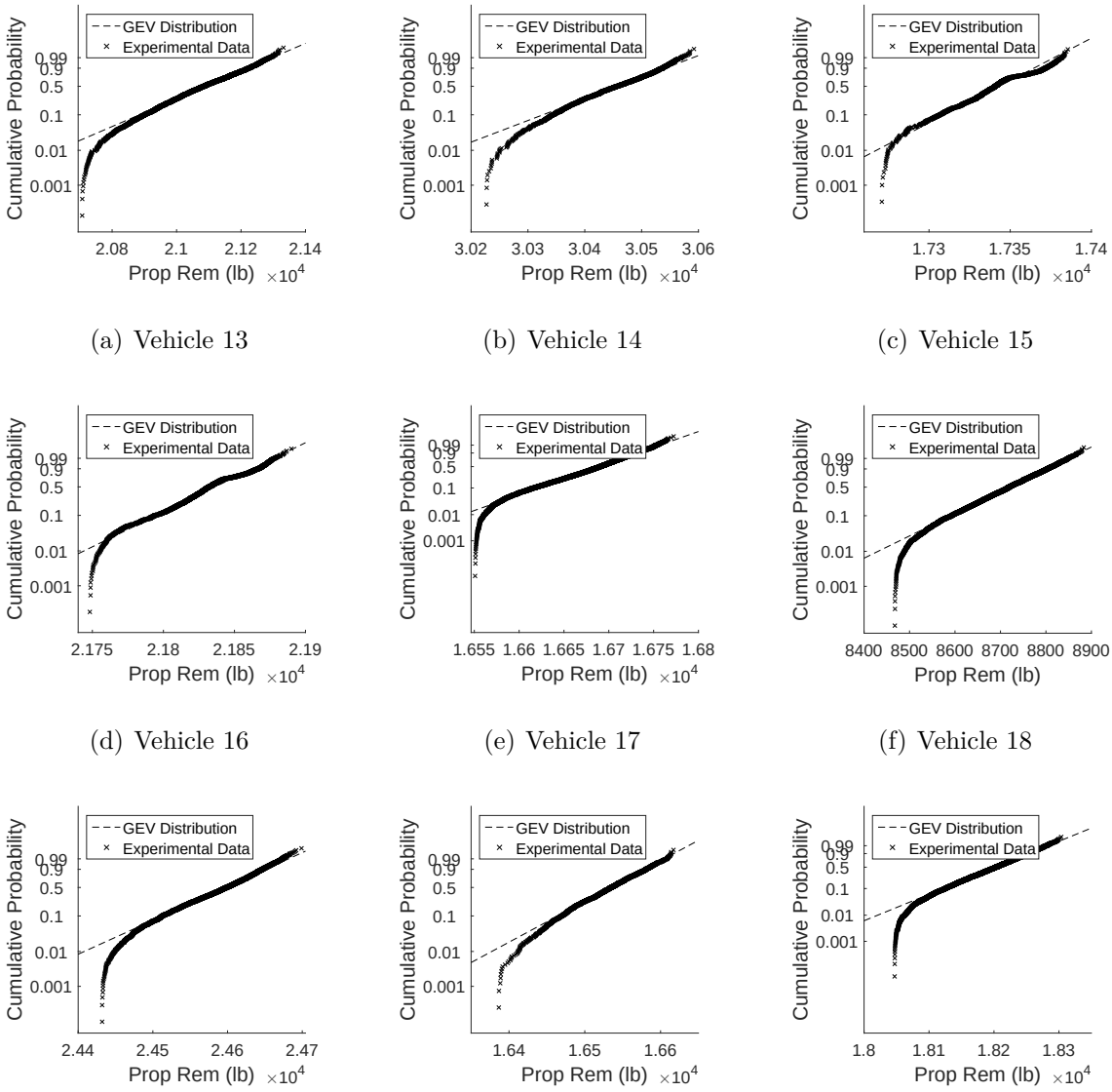


(k) Vehicle 11



(l) Vehicle 12

Figure 92: Probability plots for repeated optimization analyses for vehicles 1 - 12



(a) Vehicle 13 (b) Vehicle 14 (c) Vehicle 15

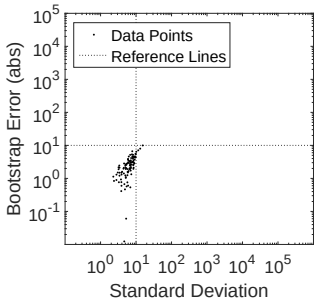
(d) Vehicle 16 (e) Vehicle 17 (f) Vehicle 18

(g) Vehicle 19 (h) Vehicle 20 (i) Vehicle 21

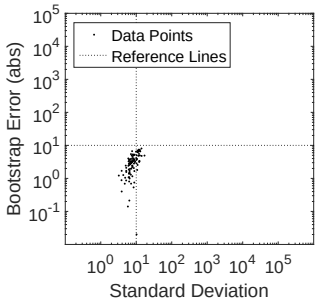
Figure 93: Probability plots for repeated optimization analyses for vehicles 13 - 21

D.3 Bootstrap Method Plots

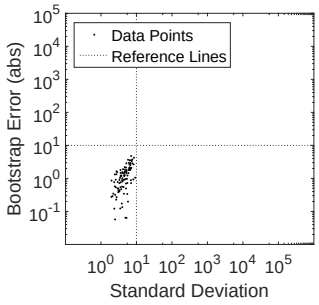
This appendix section includes plots similar to those shown in Figures 68 and 69 and Figures 70 and 71 in Section 3.5.3. In Section 3.5.3 the bootstrap method was repeatedly applied using a sample size of 25. Here, the same process is applied using different sample sizes. Figures 94 and 95 show the bootstrap error vs standard deviation and Figures 96 and 97 show the bootstrap error vs the sample error for all the representative vehicles when the sample size is set to 50. Figures 98 through 101 show the same plots when the sample size is set to 100 and Figures 102 through 105 show the results when the sample size is set to 200.



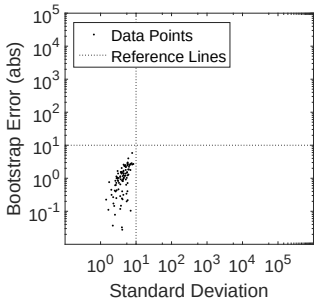
(a) Vehicle 1



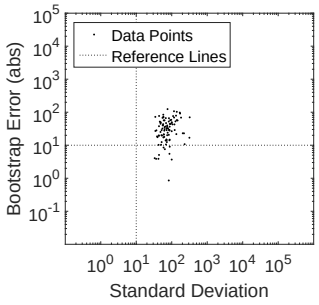
(b) Vehicle 2



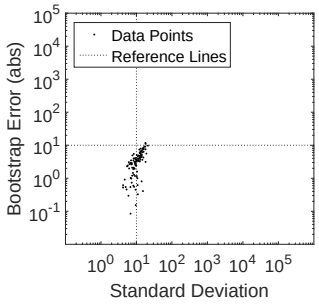
(c) Vehicle 3



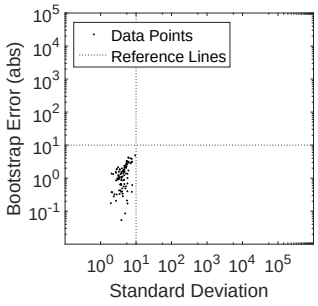
(d) Vehicle 4



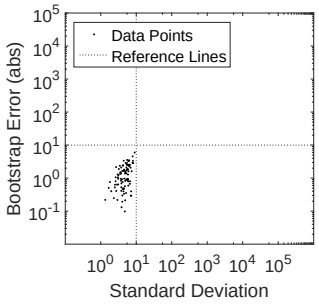
(e) Vehicle 5



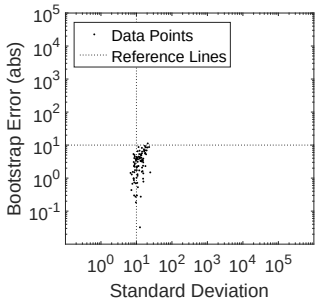
(f) Vehicle 6



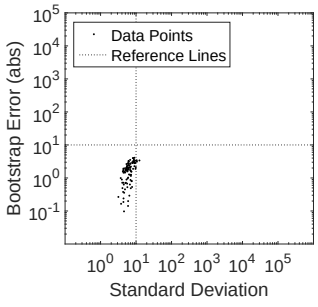
(g) Vehicle 7



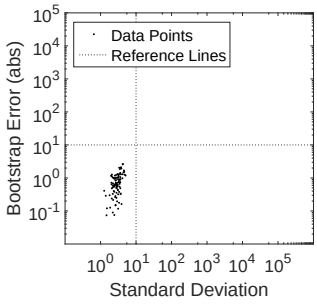
(h) Vehicle 8



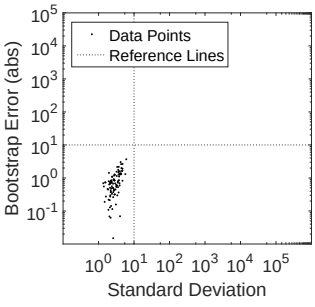
(i) Vehicle 9



(j) Vehicle 10

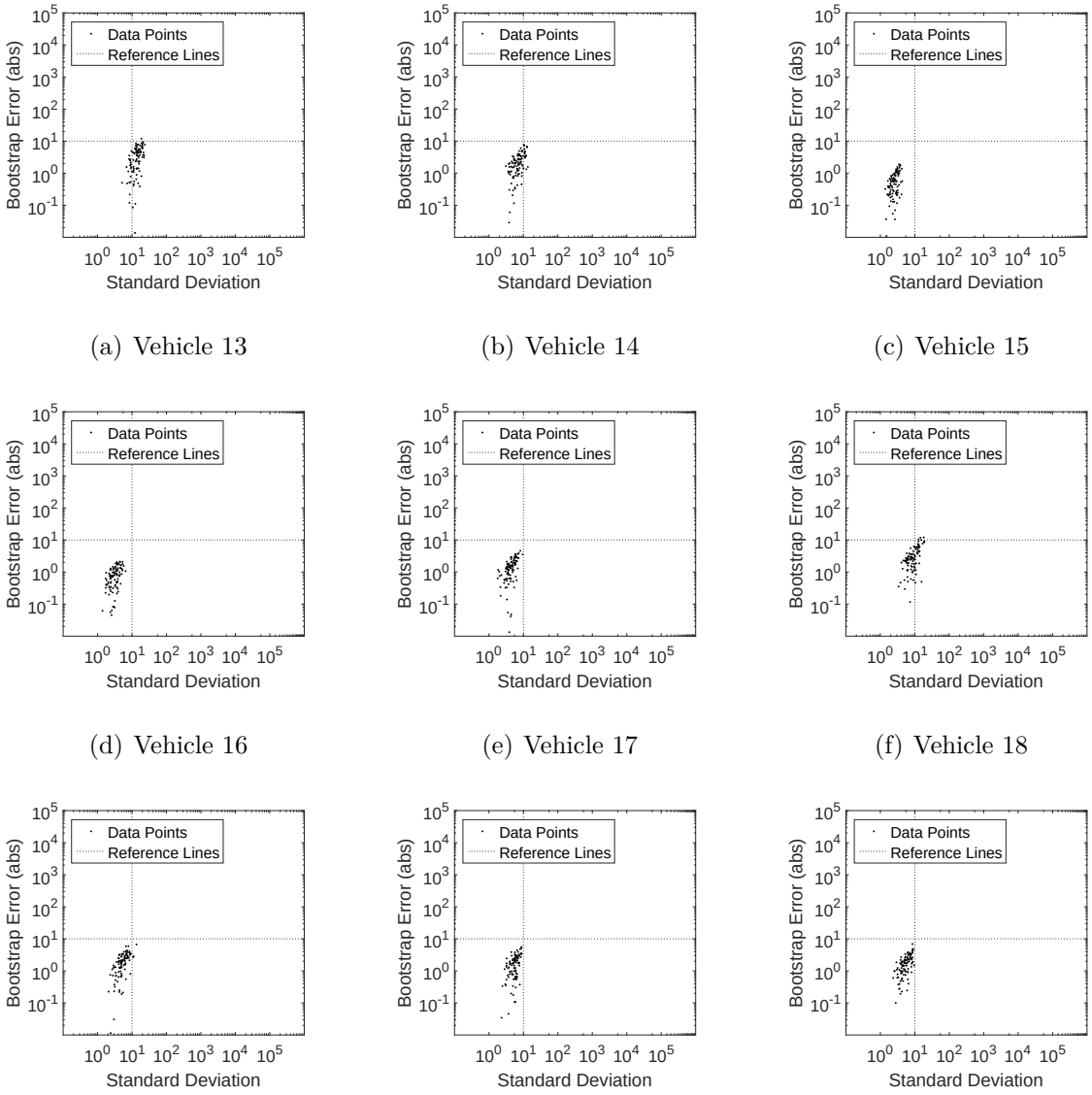


(k) Vehicle 11



(l) Vehicle 12

Figure 94: Bootstrap error vs standard deviation for vehicles 1-12 using a sample size of 50

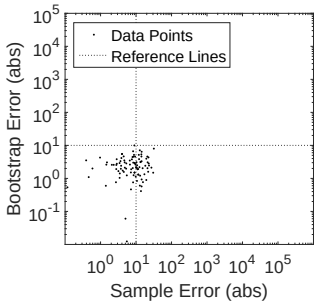


(a) Vehicle 13 (b) Vehicle 14 (c) Vehicle 15

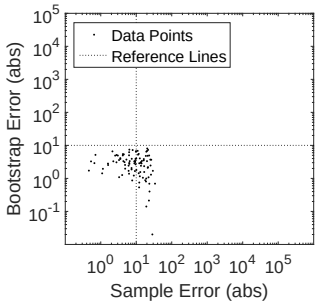
(d) Vehicle 16 (e) Vehicle 17 (f) Vehicle 18

(g) Vehicle 19 (h) Vehicle 20 (i) Vehicle 21

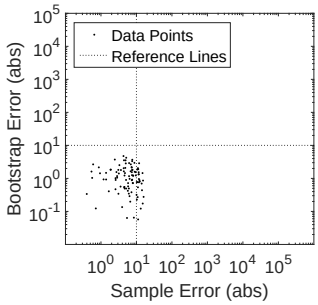
Figure 95: Bootstrap error vs standard deviation for vehicles 13-21 using a sample size of 50



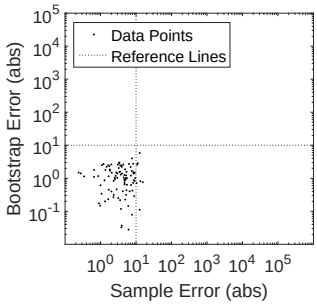
(a) Vehicle 1



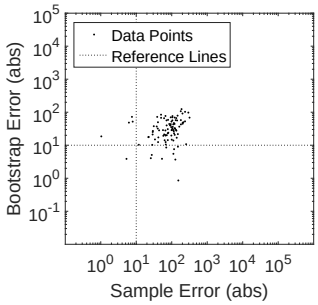
(b) Vehicle 2



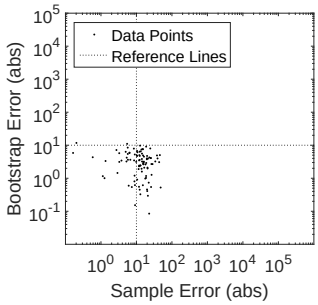
(c) Vehicle 3



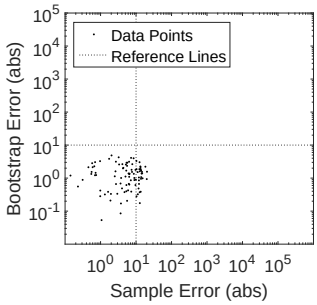
(d) Vehicle 4



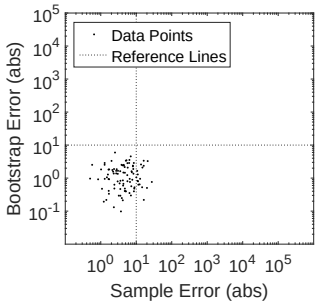
(e) Vehicle 5



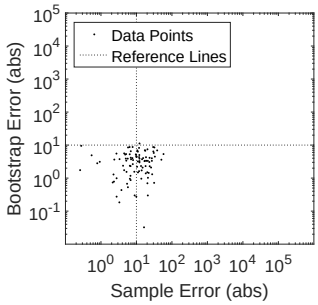
(f) Vehicle 6



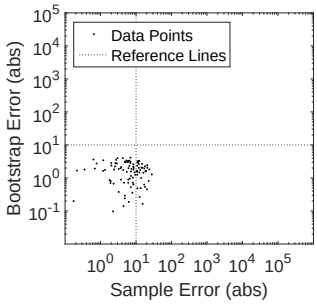
(g) Vehicle 7



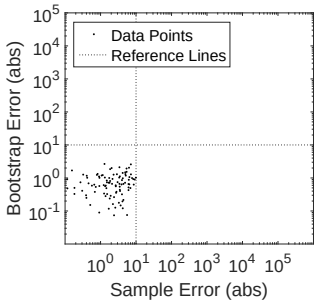
(h) Vehicle 8



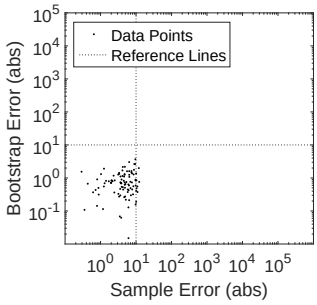
(i) Vehicle 9



(j) Vehicle 10

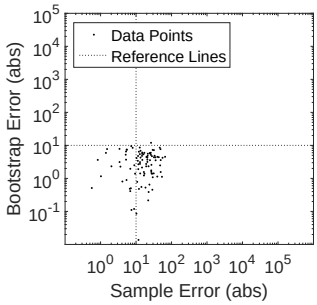


(k) Vehicle 11

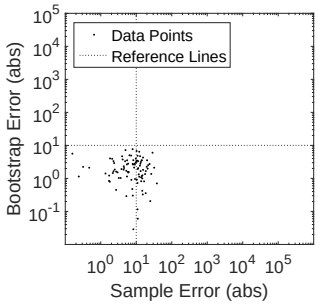


(l) Vehicle 12

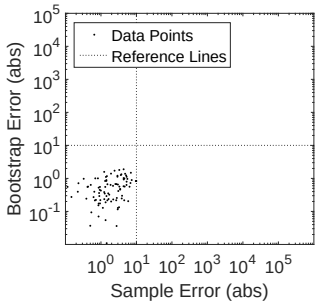
Figure 96: Bootstrap error vs sample error for vehicles 1-12 using a sample size of 50



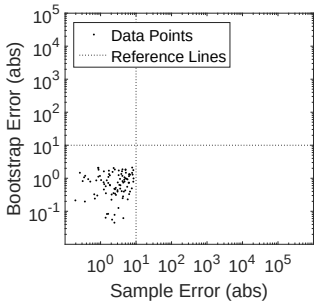
(a) Vehicle 13



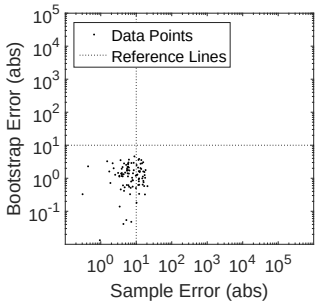
(b) Vehicle 14



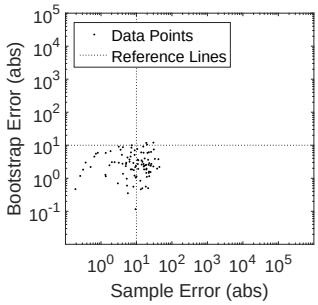
(c) Vehicle 15



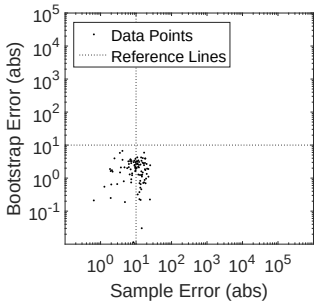
(d) Vehicle 16



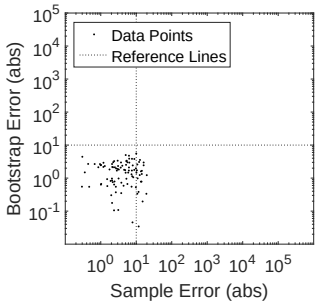
(e) Vehicle 17



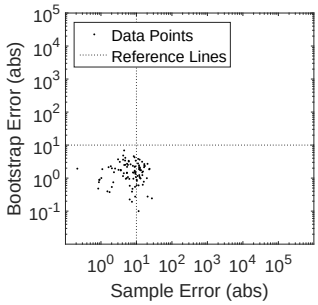
(f) Vehicle 18



(g) Vehicle 19

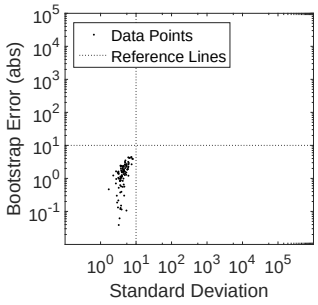


(h) Vehicle 20

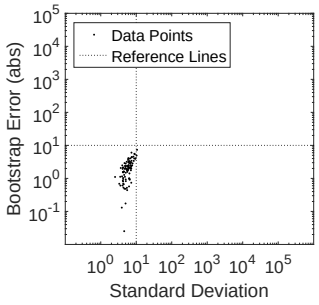


(i) Vehicle 21

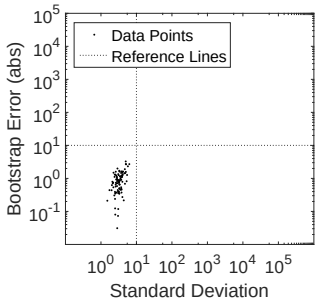
Figure 97: Bootstrap error vs sample error for vehicles 13-21 using a sample size of 50



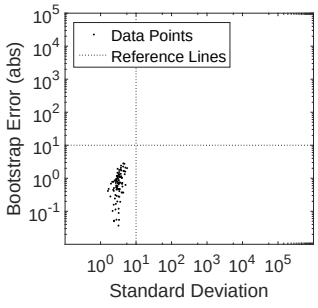
(a) Vehicle 1



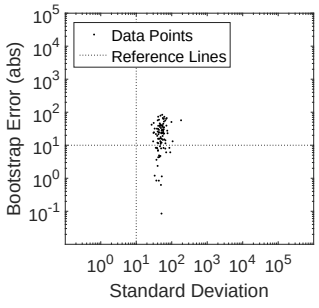
(b) Vehicle 2



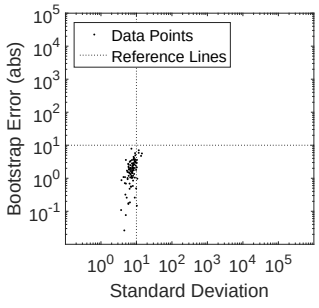
(c) Vehicle 3



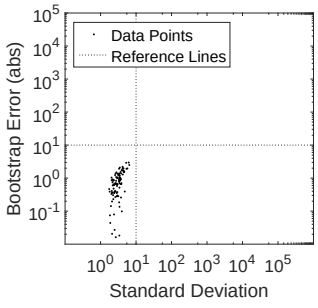
(d) Vehicle 4



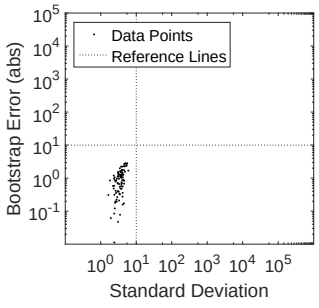
(e) Vehicle 5



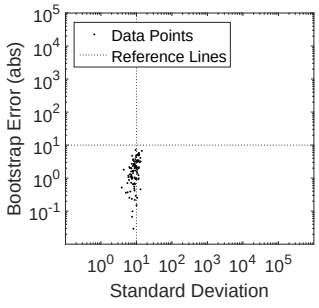
(f) Vehicle 6



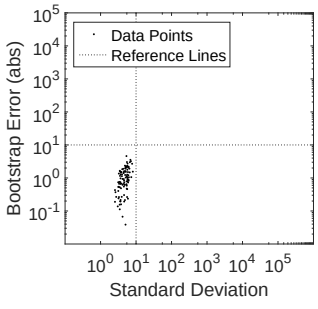
(g) Vehicle 7



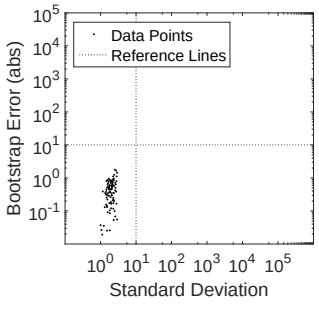
(h) Vehicle 8



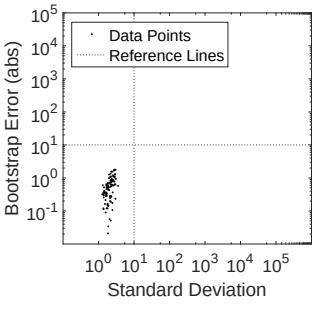
(i) Vehicle 9



(j) Vehicle 10

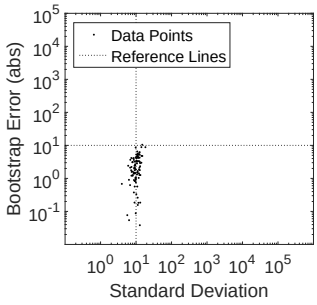


(k) Vehicle 11

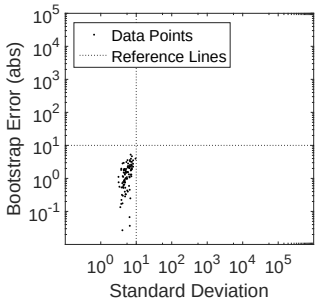


(l) Vehicle 12

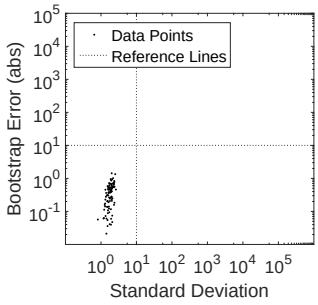
Figure 98: Bootstrap error vs standard deviation for vehicles 1-12 using a sample size of 100



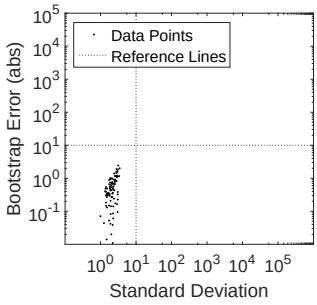
(a) Vehicle 13



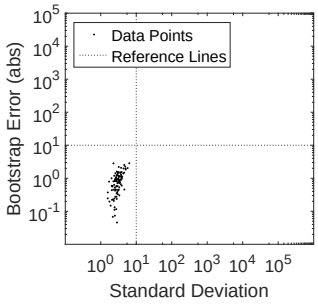
(b) Vehicle 14



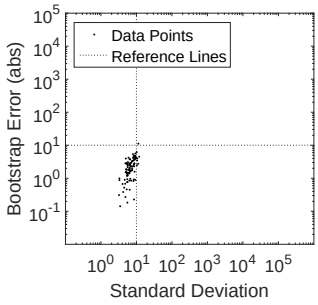
(c) Vehicle 15



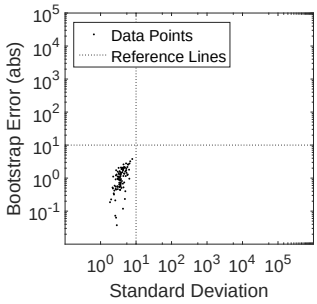
(d) Vehicle 16



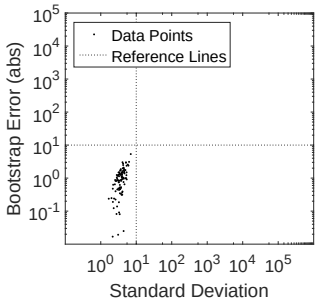
(e) Vehicle 17



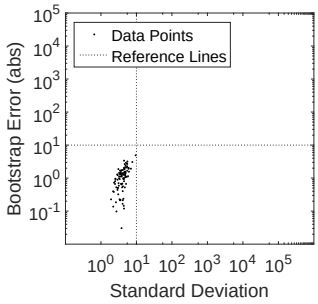
(f) Vehicle 18



(g) Vehicle 19

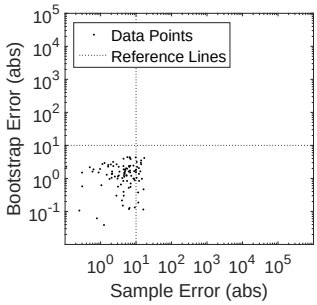


(h) Vehicle 20

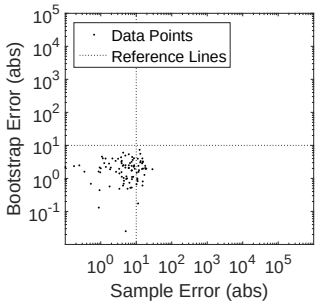


(i) Vehicle 21

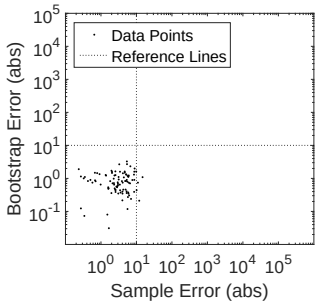
Figure 99: Bootstrap error vs standard deviation for vehicles 13-21 using a sample size of 100



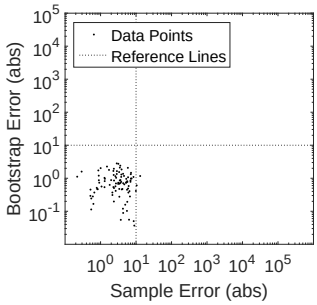
(a) Vehicle 1



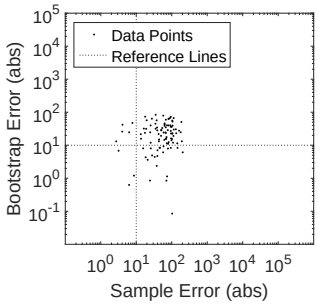
(b) Vehicle 2



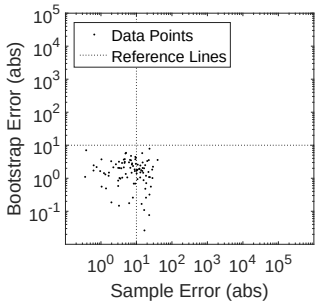
(c) Vehicle 3



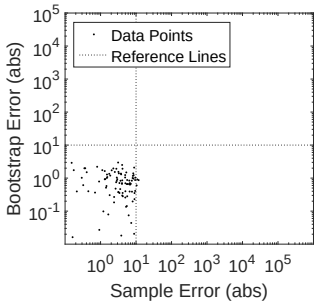
(d) Vehicle 4



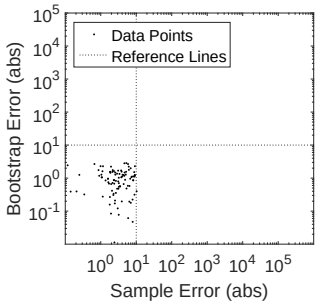
(e) Vehicle 5



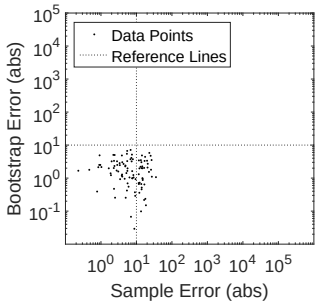
(f) Vehicle 6



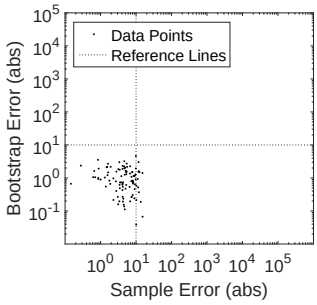
(g) Vehicle 7



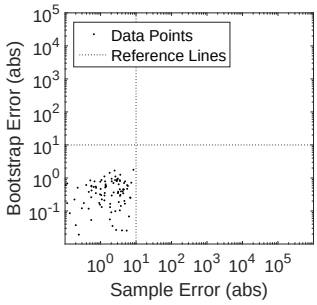
(h) Vehicle 8



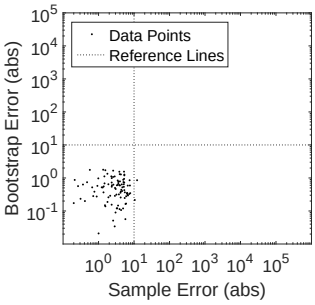
(i) Vehicle 9



(j) Vehicle 10

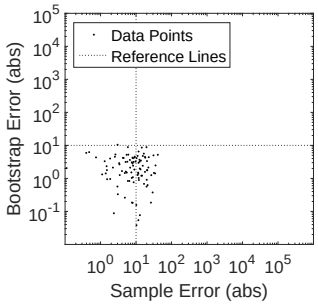


(k) Vehicle 11

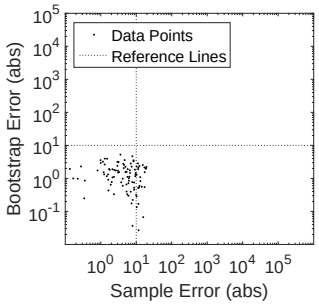


(l) Vehicle 12

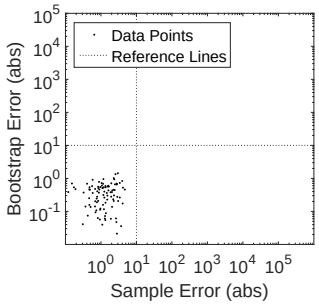
Figure 100: Bootstrap error vs sample error for vehicles 1-12 using a sample size of 100



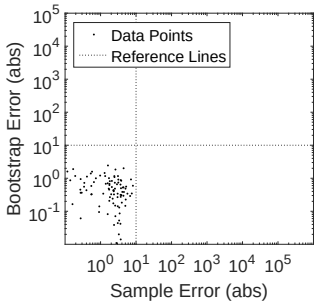
(a) Vehicle 13



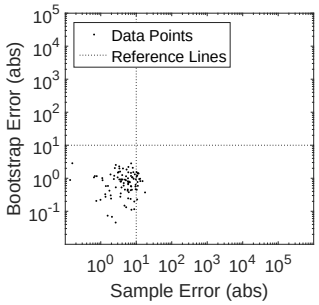
(b) Vehicle 14



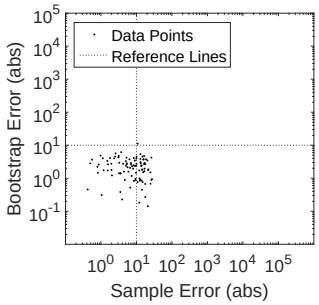
(c) Vehicle 15



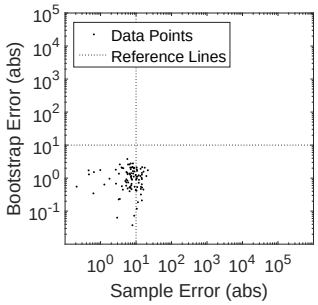
(d) Vehicle 16



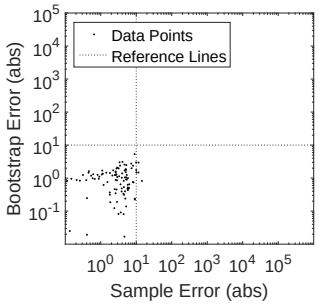
(e) Vehicle 17



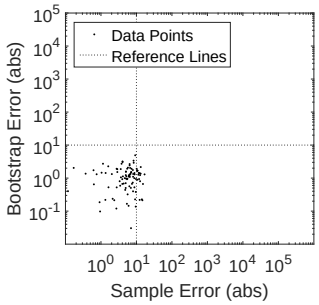
(f) Vehicle 18



(g) Vehicle 19

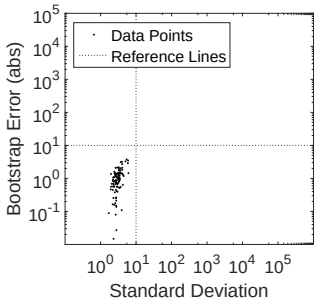


(h) Vehicle 20

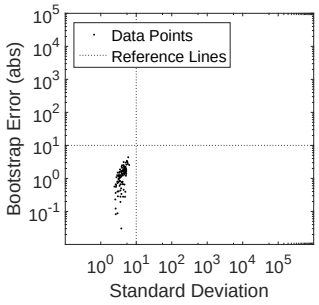


(i) Vehicle 21

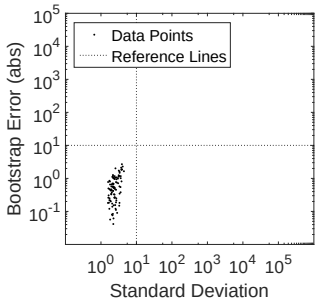
Figure 101: Bootstrap error vs sample error for vehicles 13-21 using a sample size of 100



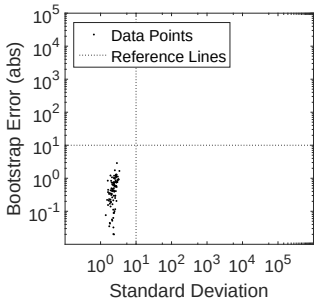
(a) Vehicle 1



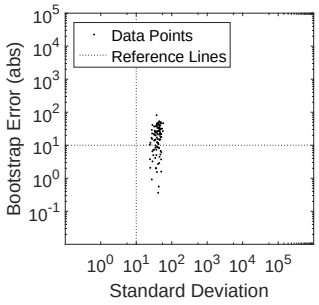
(b) Vehicle 2



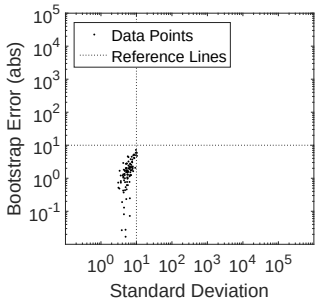
(c) Vehicle 3



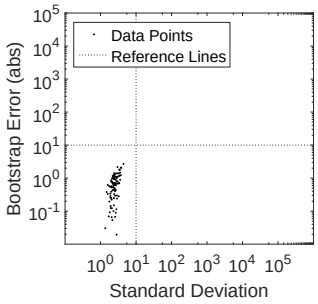
(d) Vehicle 4



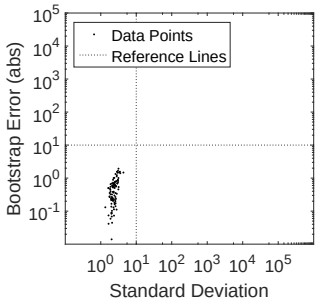
(e) Vehicle 5



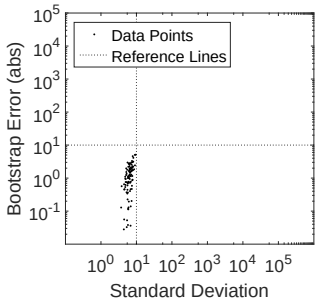
(f) Vehicle 6



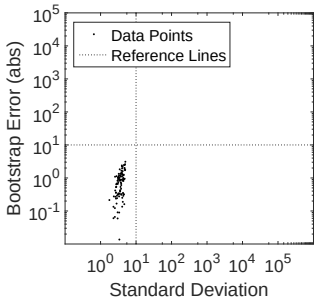
(g) Vehicle 7



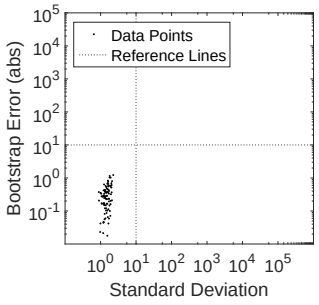
(h) Vehicle 8



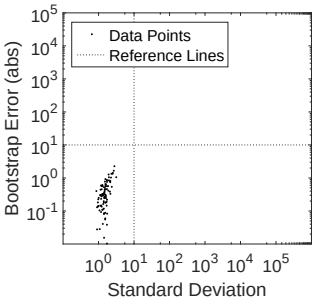
(i) Vehicle 9



(j) Vehicle 10

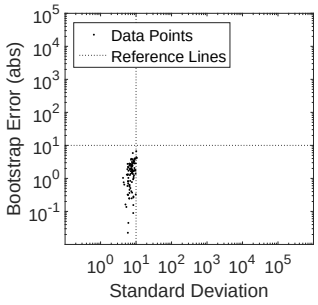


(k) Vehicle 11

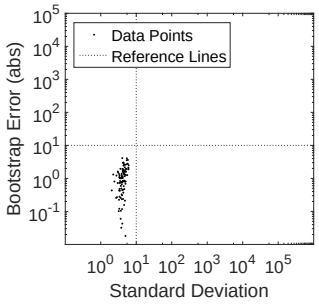


(l) Vehicle 12

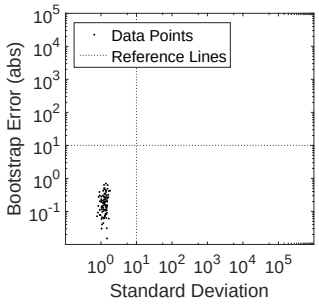
Figure 102: Bootstrap error vs standard deviation for vehicles 1-12 using a sample size of 200



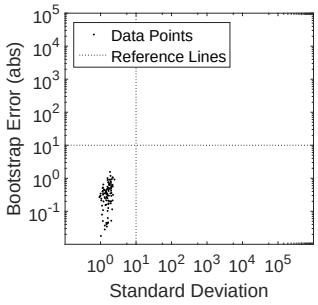
(a) Vehicle 13



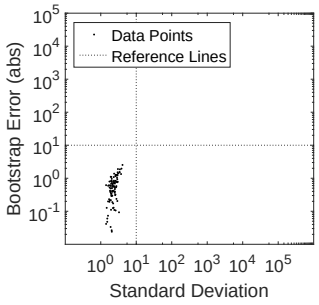
(b) Vehicle 14



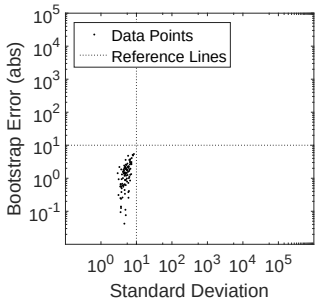
(c) Vehicle 15



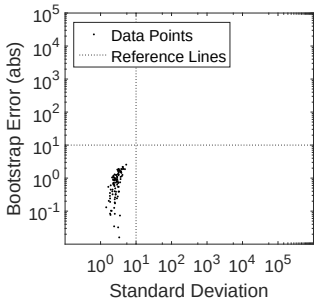
(d) Vehicle 16



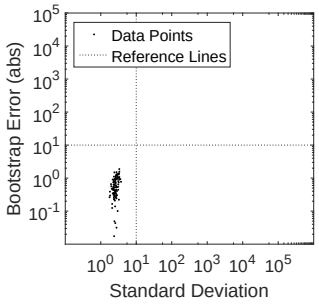
(e) Vehicle 17



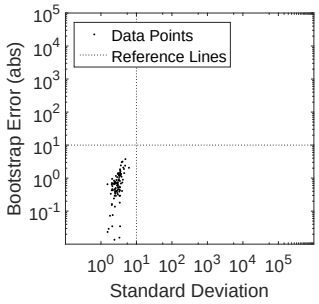
(f) Vehicle 18



(g) Vehicle 19

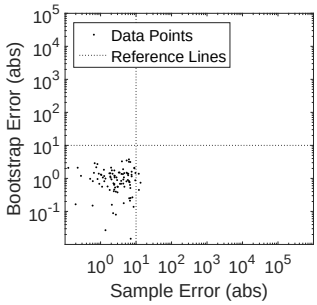


(h) Vehicle 20

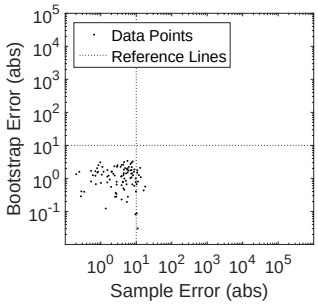


(i) Vehicle 21

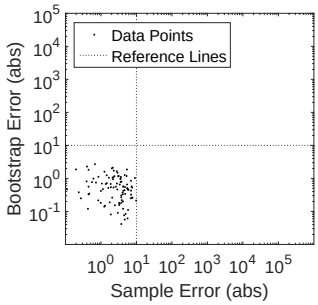
Figure 103: Bootstrap error vs standard deviation for vehicles 13-21 using a sample size of 200



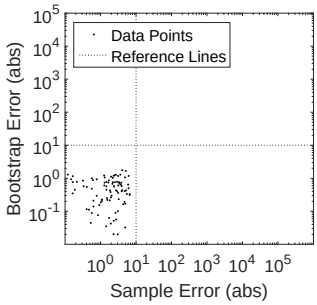
(a) Vehicle 1



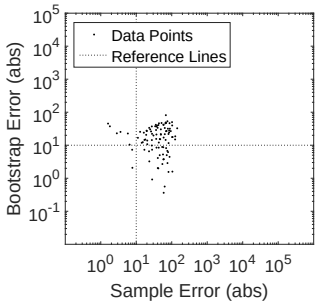
(b) Vehicle 2



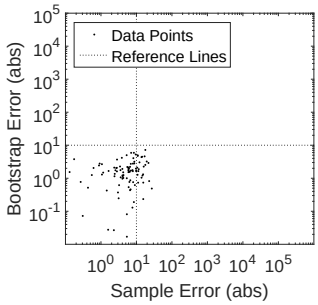
(c) Vehicle 3



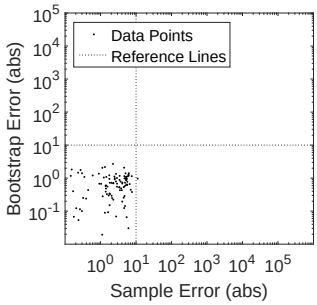
(d) Vehicle 4



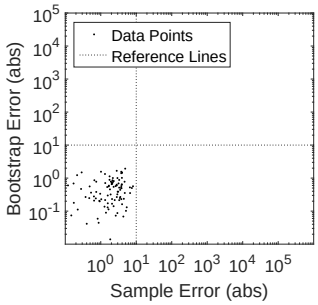
(e) Vehicle 5



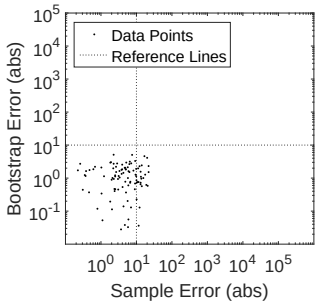
(f) Vehicle 6



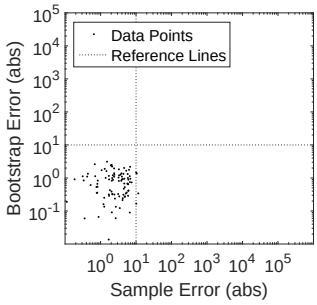
(g) Vehicle 7



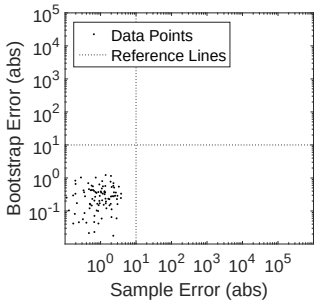
(h) Vehicle 8



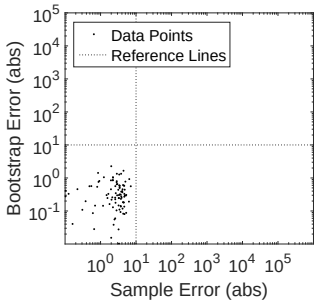
(i) Vehicle 9



(j) Vehicle 10

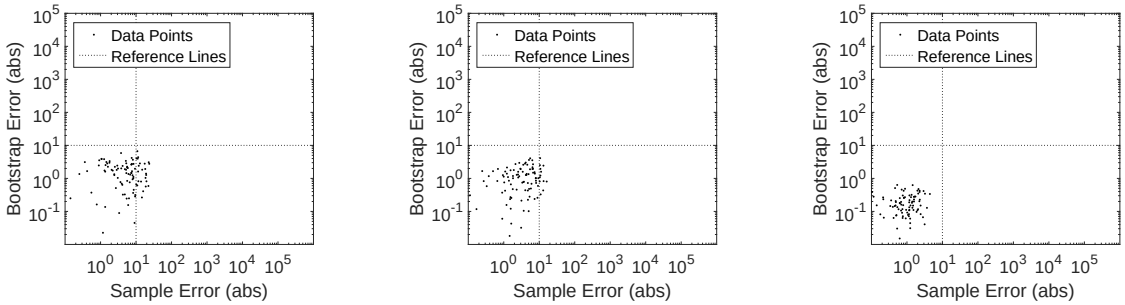


(k) Vehicle 11

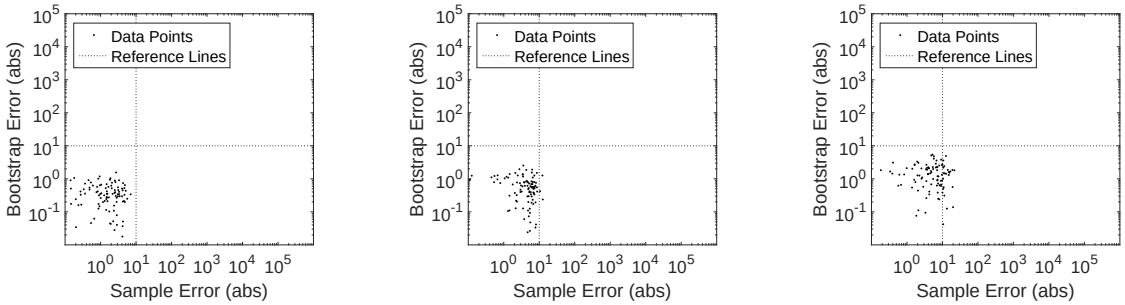


(l) Vehicle 12

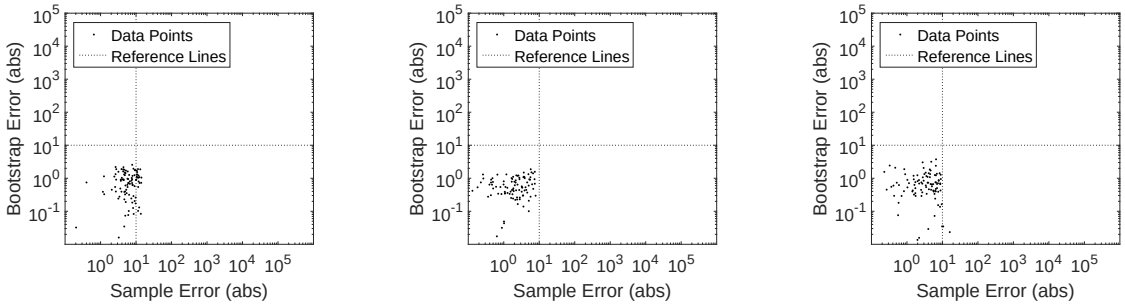
Figure 104: Bootstrap error vs sample error for vehicles 1-12 using a sample size of 200



(a) Vehicle 13 (b) Vehicle 14 (c) Vehicle 15



(d) Vehicle 16 (e) Vehicle 17 (f) Vehicle 18



(g) Vehicle 19 (h) Vehicle 20 (i) Vehicle 21

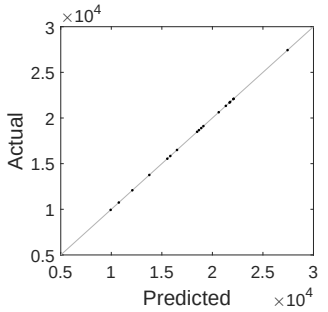
Figure 105: Bootstrap error vs sample error for vehicles 13-21 using a sample size of 200

D.4 Surrogate Model Plots

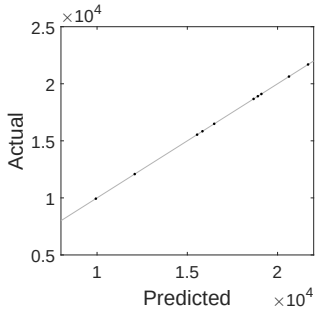
This appendix section includes the actual by predicted and residual by predicted plots for the surrogate models fit for both fit and validation data. The caption for each subplot indicates whether fit or validation data is shown and the number of required repetitions. The number of cases used, reported in the caption for each figure, refers to the initial number of cases in the DOE. The actual number of usable cases is given in Table 20 in Section 3.7.5.

D.4.1 Neural Networks

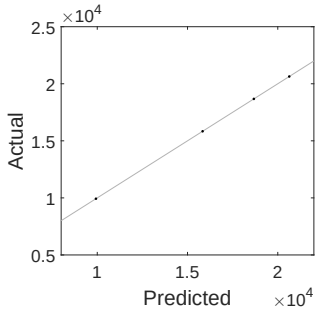
Figures 106 through 112 show the results for the neural networks. Each figure represents a neural network generated using an LHC design with the number of cases reported in the caption. The sub-figures (a)-(f) show the actual vs the predicted values. The data will lie in a diagonal line for a perfect fit. The sub-figures (g)-(l) show the residual by predicted. The labels on the sub-figures represent the number of repetitions required for each data point, 25, 50, or 100, and whether the figure is showing fit data or validation data. The validation plots represented with "n/a" signify that no validation points were available.



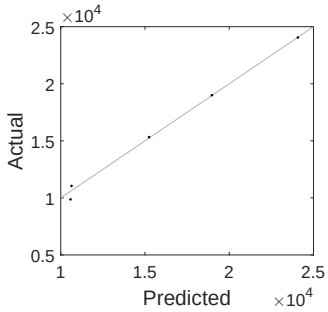
(a) Fit - 25



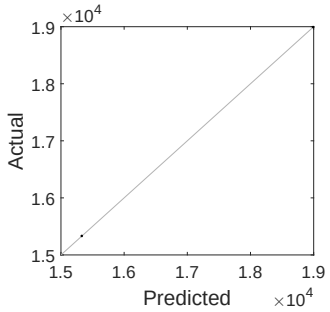
(b) Fit - 50



(c) Fit - 100



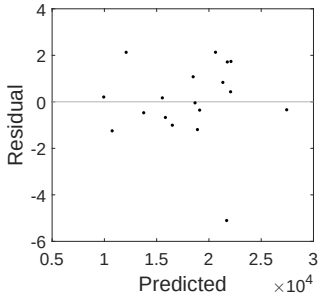
(d) Validation - 25



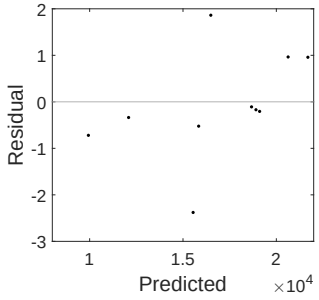
(e) Validation - 50

n/a

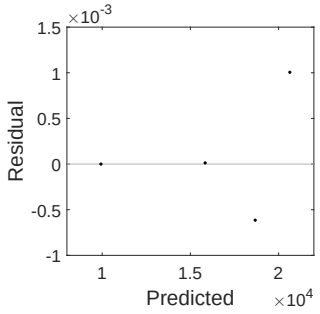
(f) Validation - 100



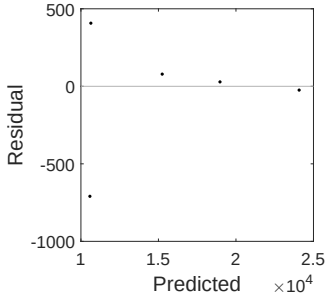
(g) Fit - 25



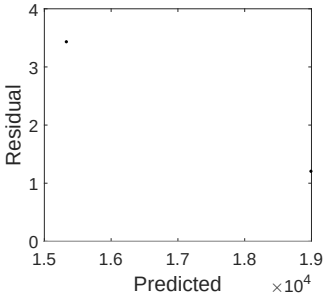
(h) Fit - 50



(i) Fit - 100



(j) Validation - 25

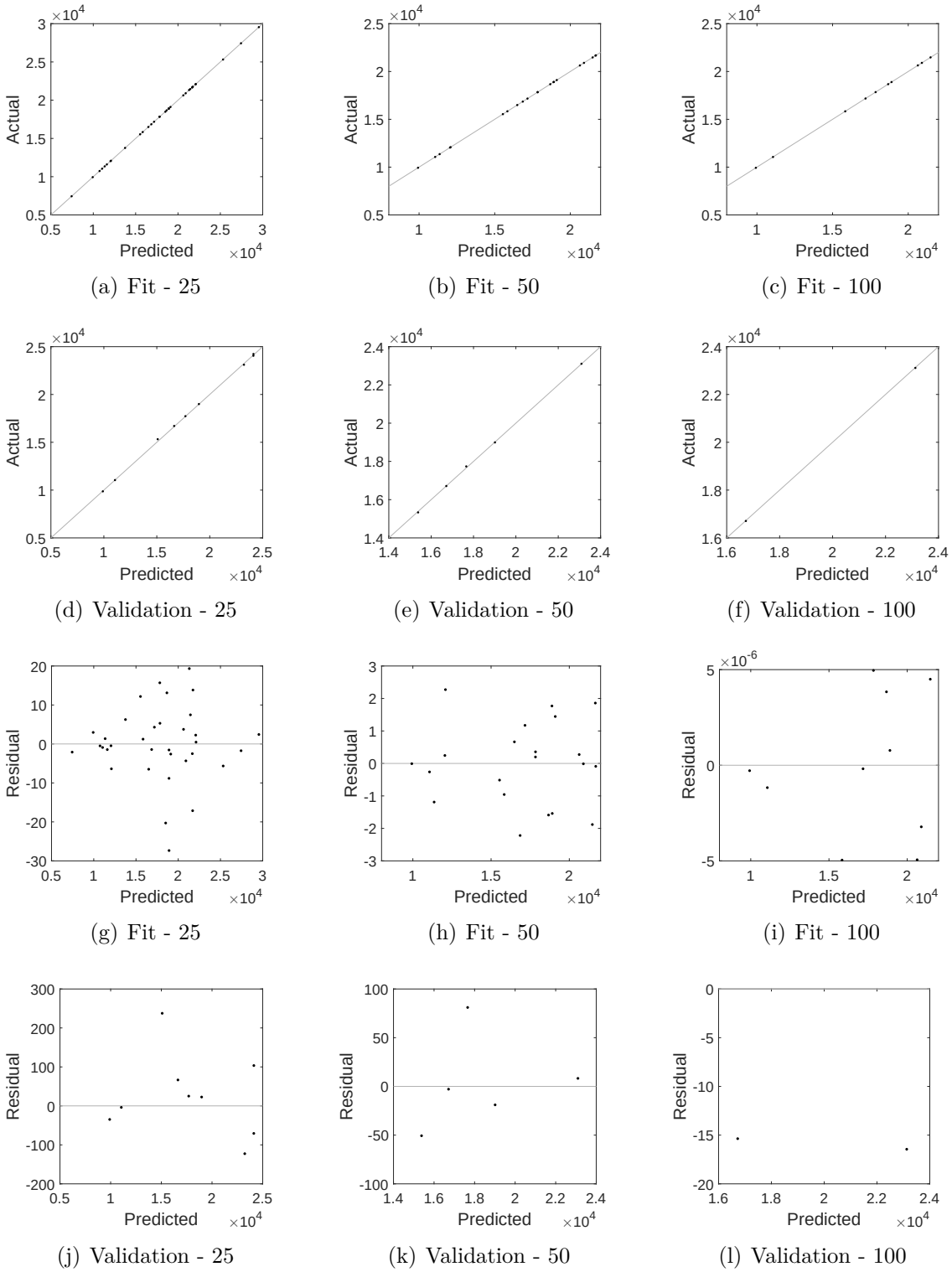


(k) Validation - 50

n/a

(l) Validation - 100

Figure 106: Goodness of fit plots for neural networks using 25 cases



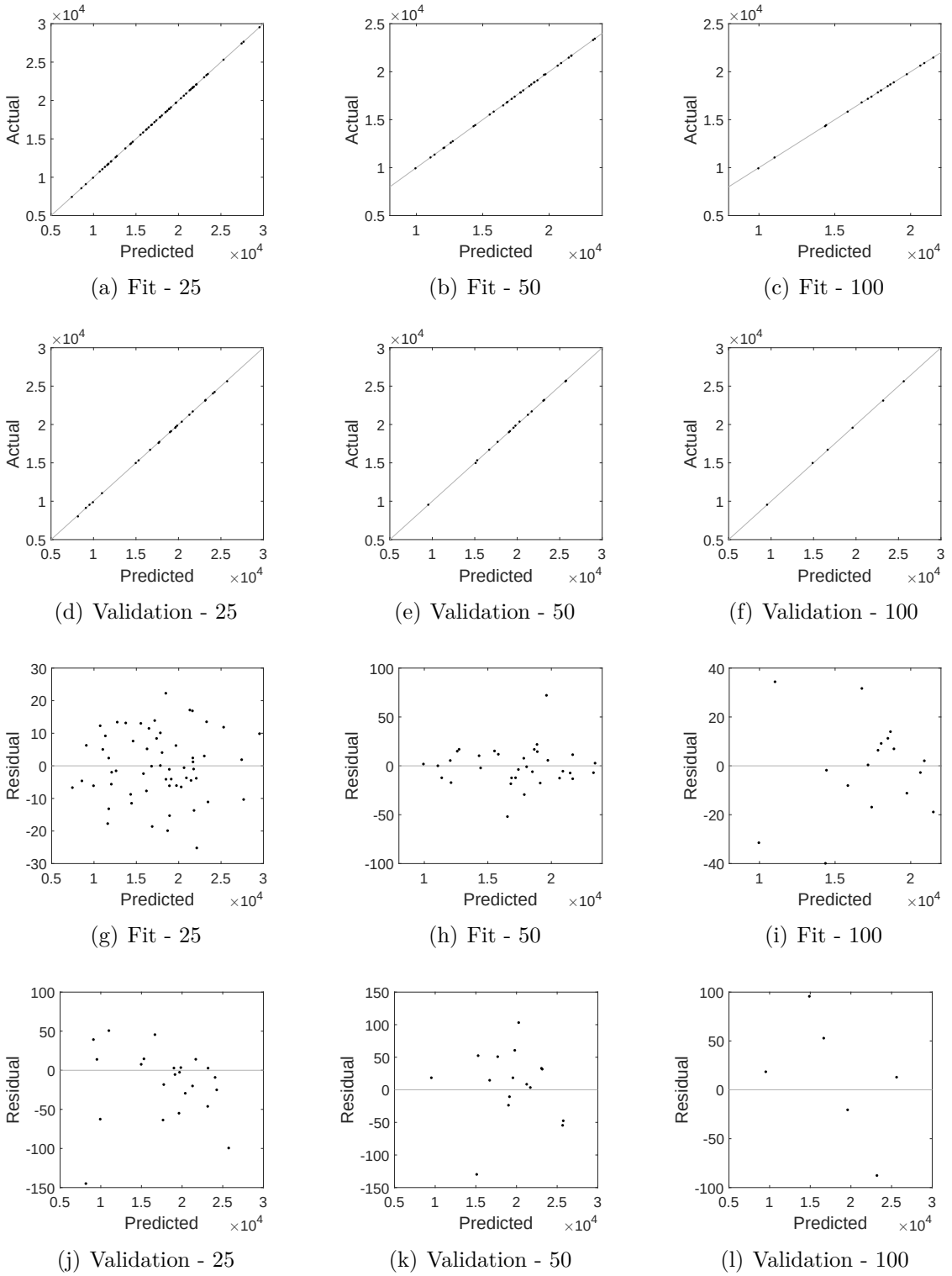
(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

Figure 107: Goodness of fit plots for neural networks using 50 cases



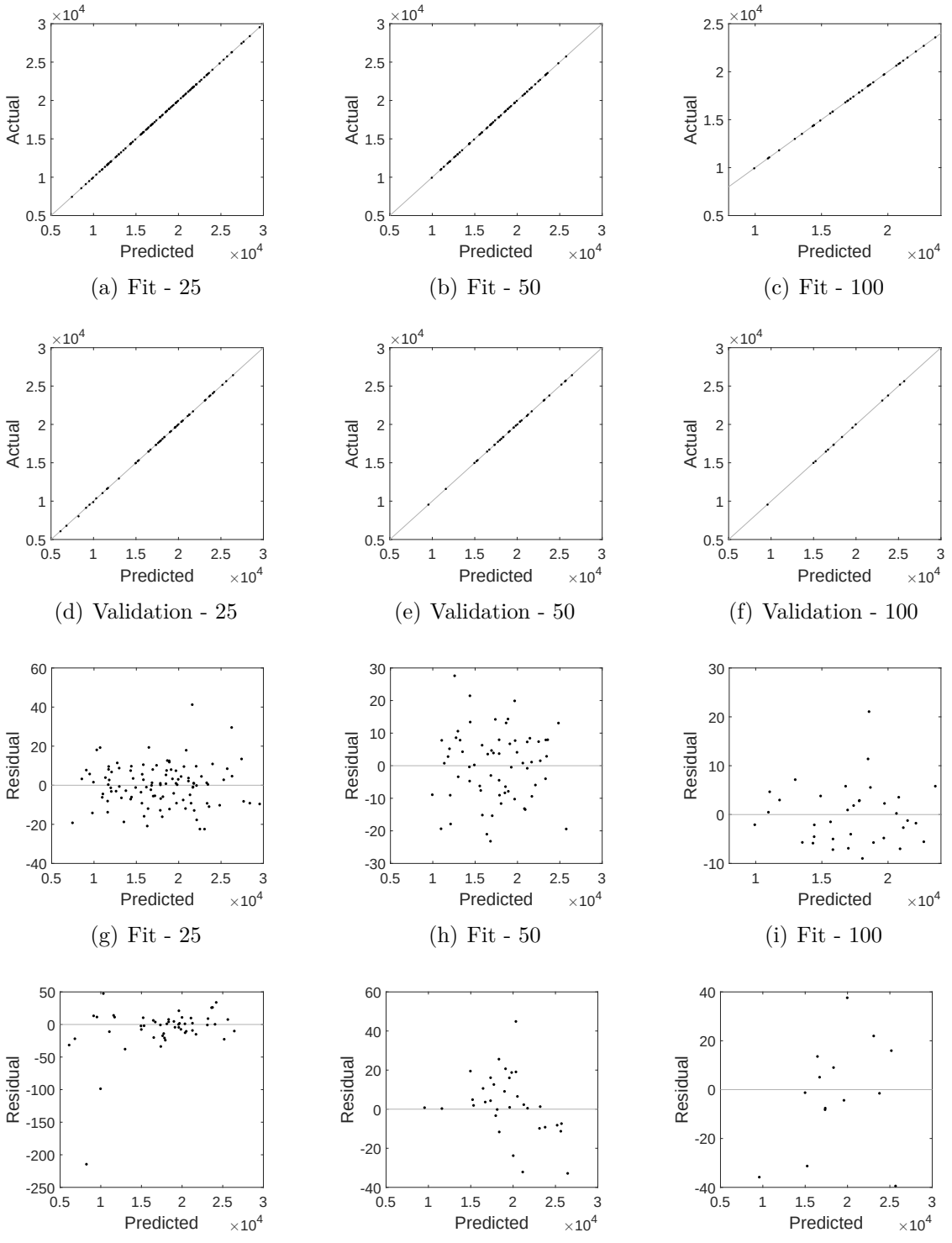
(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

Figure 108: Goodness of fit plots for neural networks using 100 cases



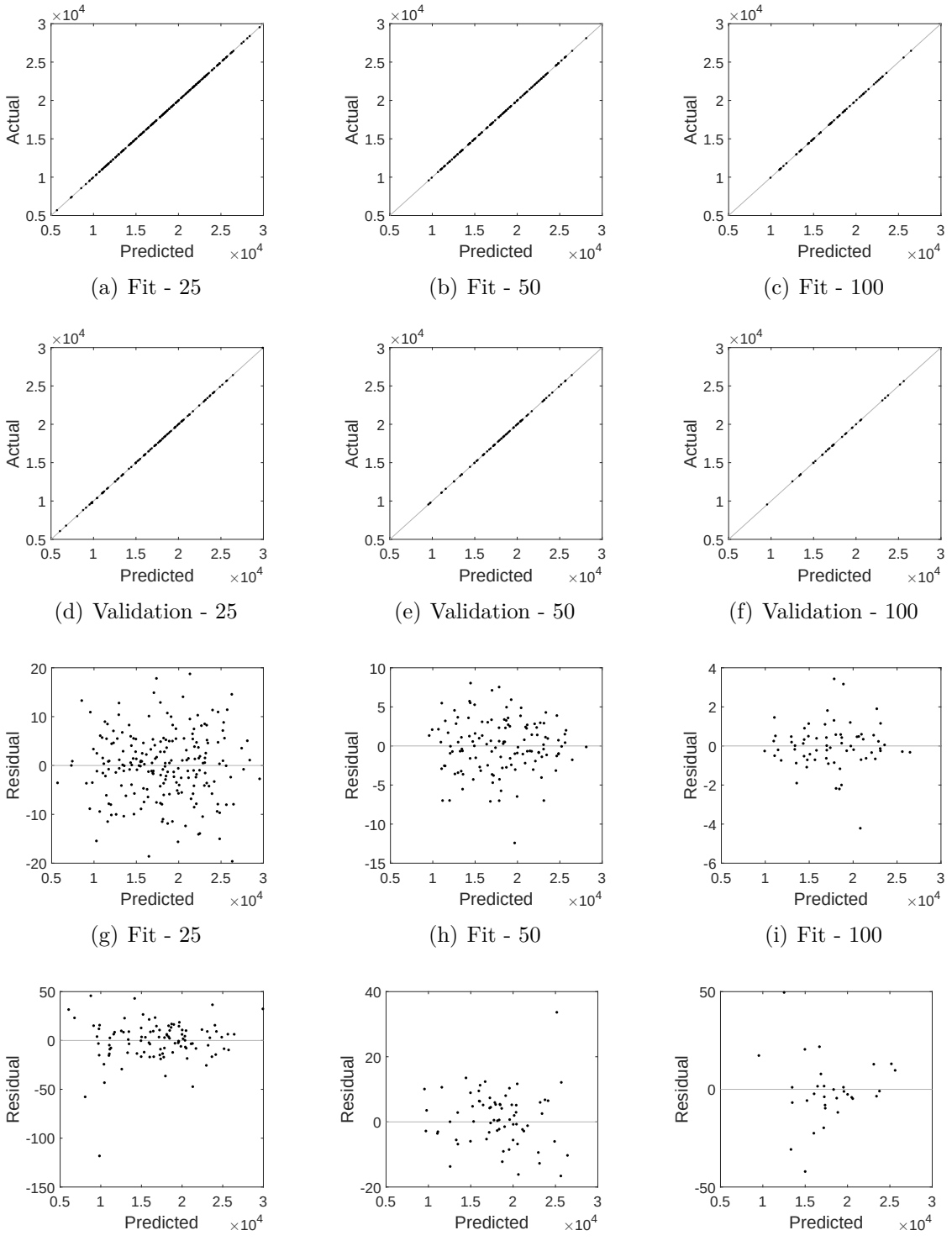
(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

Figure 109: Goodness of fit plots for neural networks using 200 cases



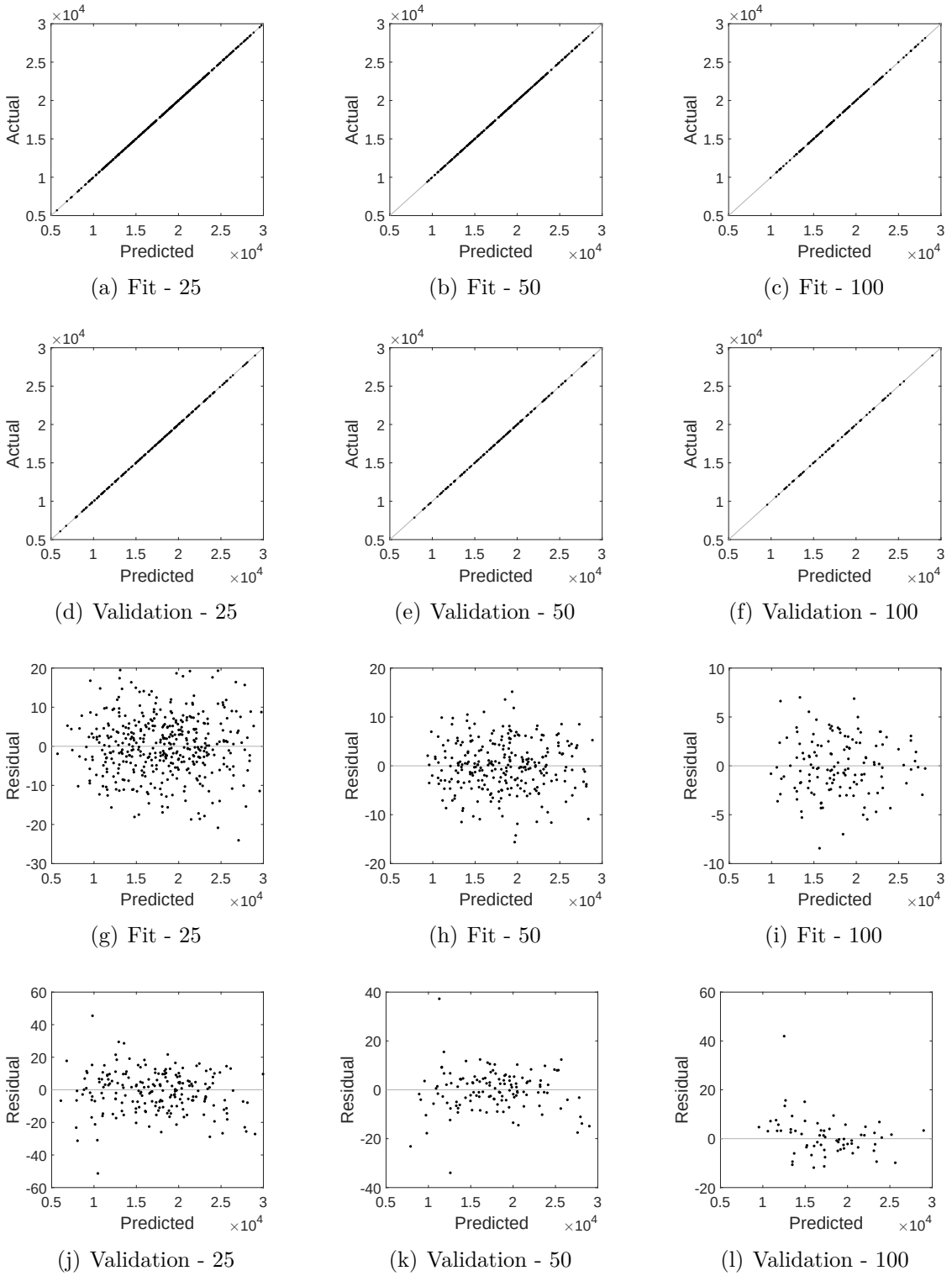
(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

Figure 110: Goodness of fit plots for neural networks using 400 cases



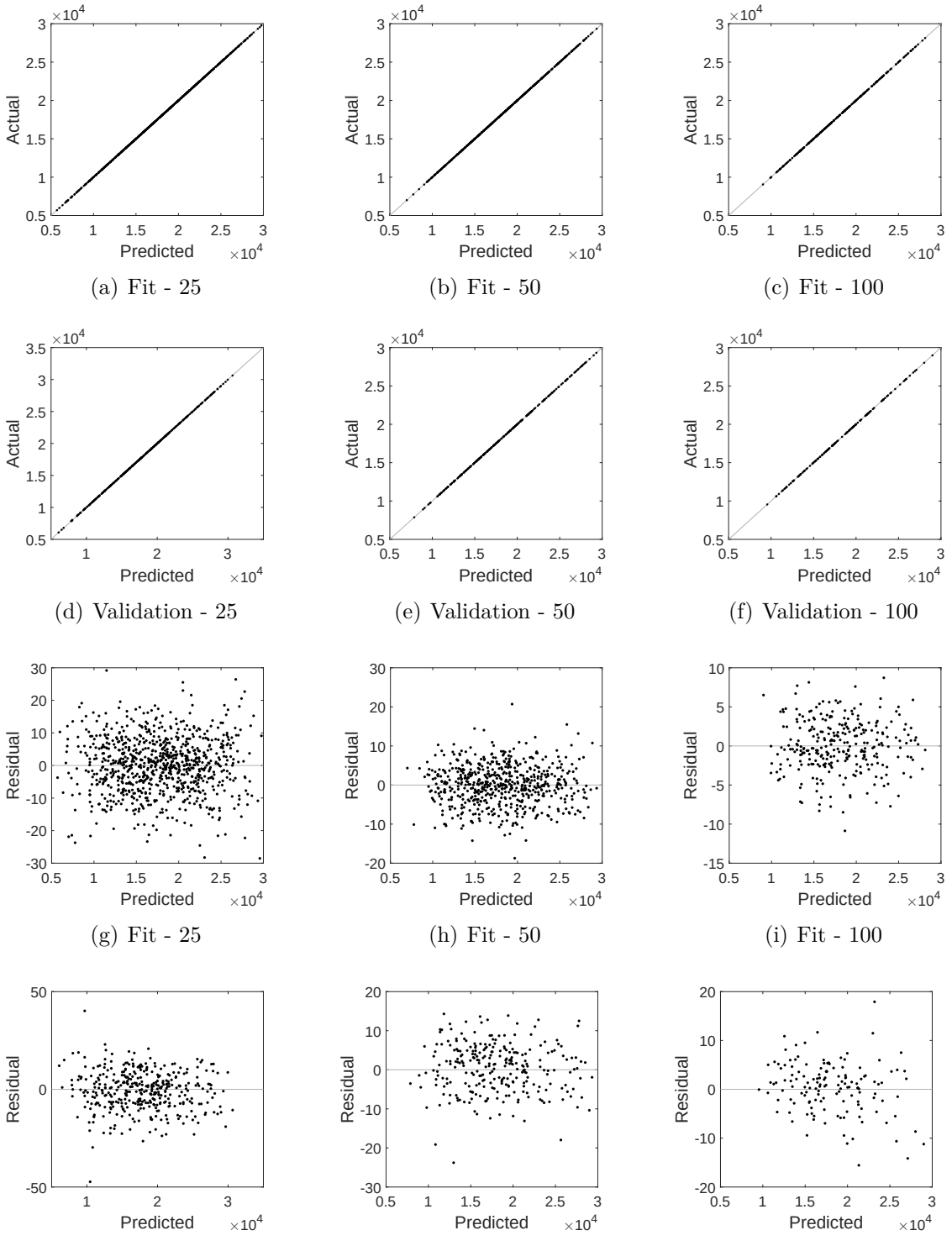
(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

Figure 111: Goodness of fit plots for neural networks using 800 cases



(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

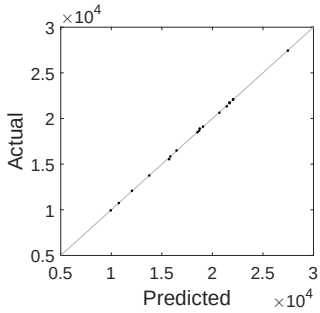
(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

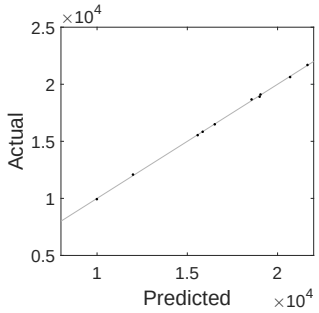
Figure 112: Goodness of fit plots for neural networks using 1600 cases

D.4.2 Polynomial Regressions

Figures 113 through 119 show the results for the polynomial regressions. As in Section D.4.1, each figure represents a regression generated using an LHC design with the number of cases reported in the caption. The sub-figures (a)-(f) show the actual vs the predicted values. The sub-figures (g)-(l) show the residual by predicted. The labels on the sub-figures represent the number of repetitions required for each data point, 25, 50, or 100, and whether the figure is showing fit data or validation data. In Figure 113 several of the sub-figures are replaced with "n/a" because there was insufficient data to generation polynomial regressions with these settings.



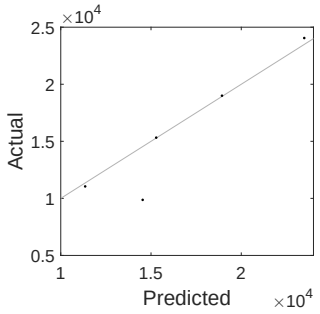
(a) Fit - 25



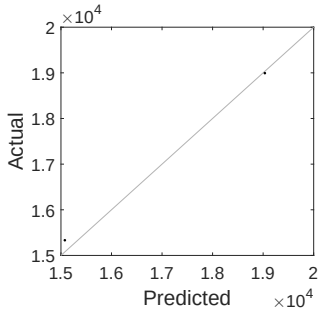
(b) Fit - 50

n/a

(c) Fit - 100



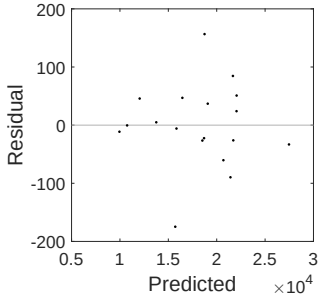
(d) Validation - 25



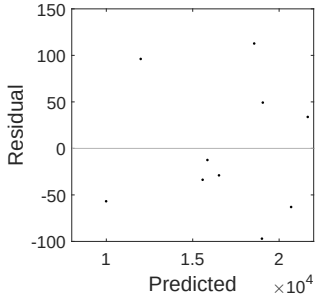
(e) Validation - 50

n/a

(f) Validation - 100



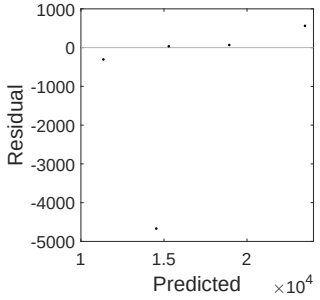
(g) Fit - 25



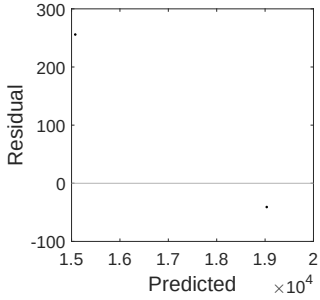
(h) Fit - 50

n/a

(i) Fit - 100



(j) Validation - 25

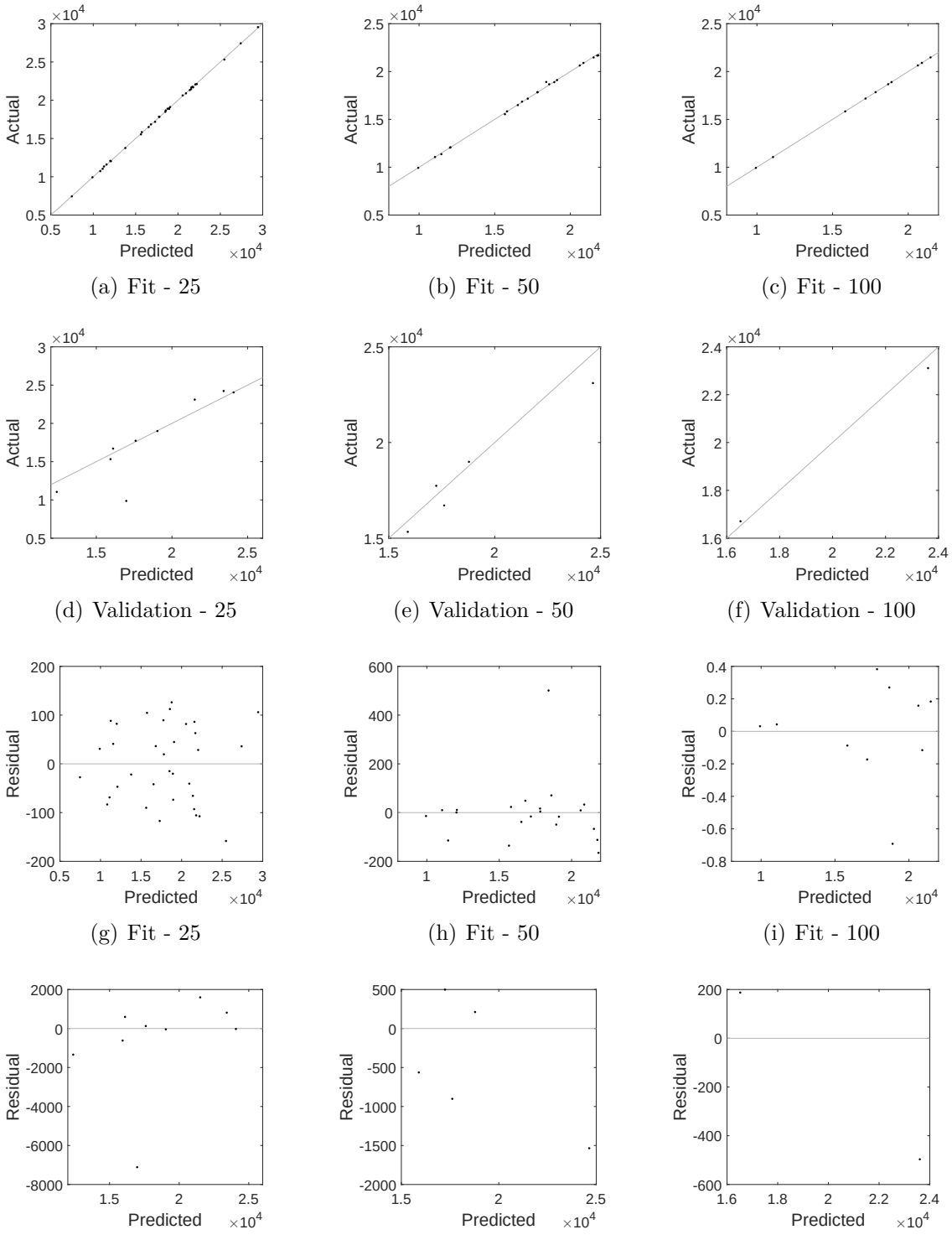


(k) Validation - 50

n/a

(l) Validation - 100

Figure 113: Goodness of fit plots for polynomial regressions using 25 cases



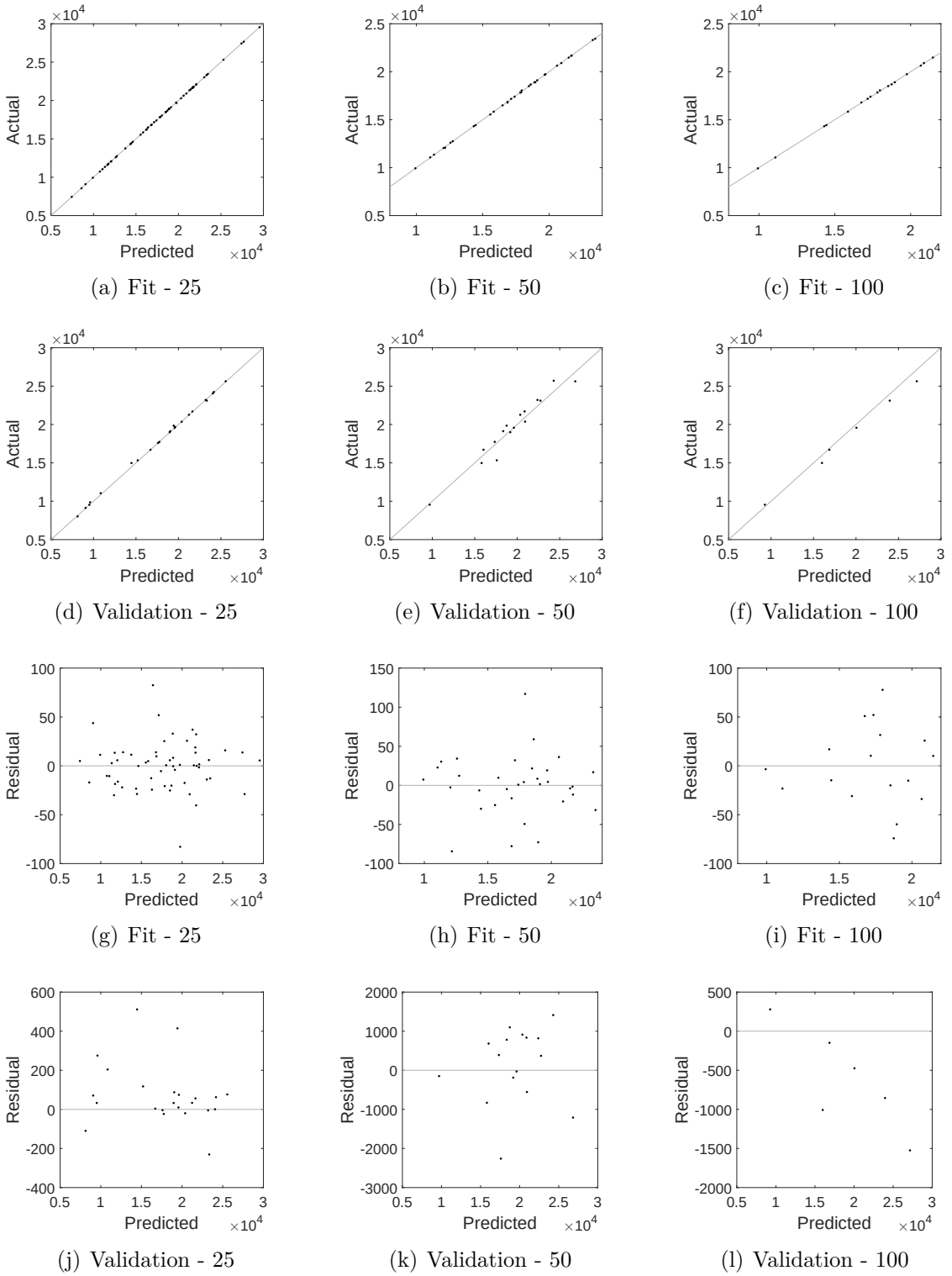
(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

Figure 114: Goodness of fit plots for polynomial regressions using 50 cases



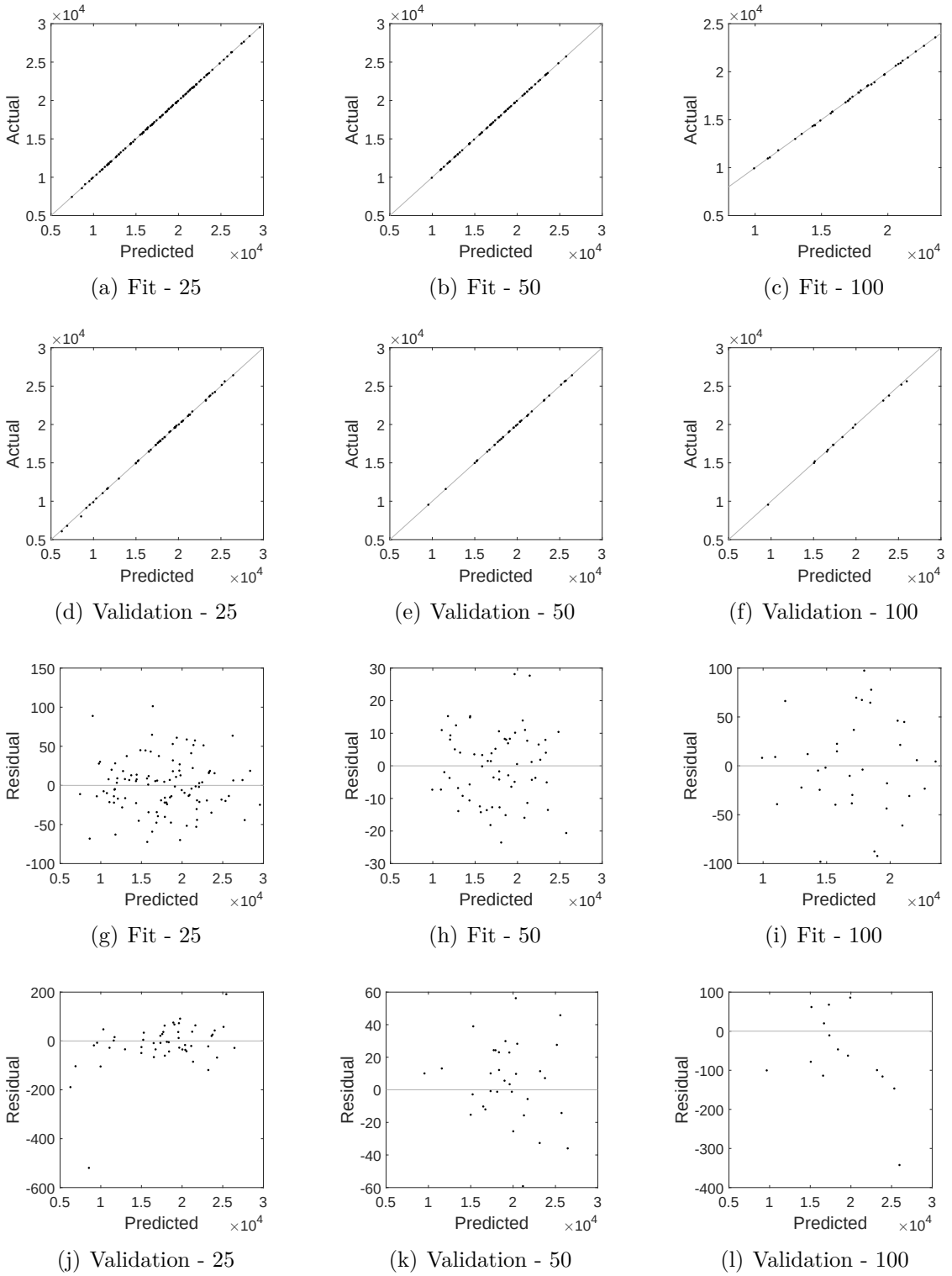
(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

Figure 115: Goodness of fit plots for polynomial regressions using 100 cases



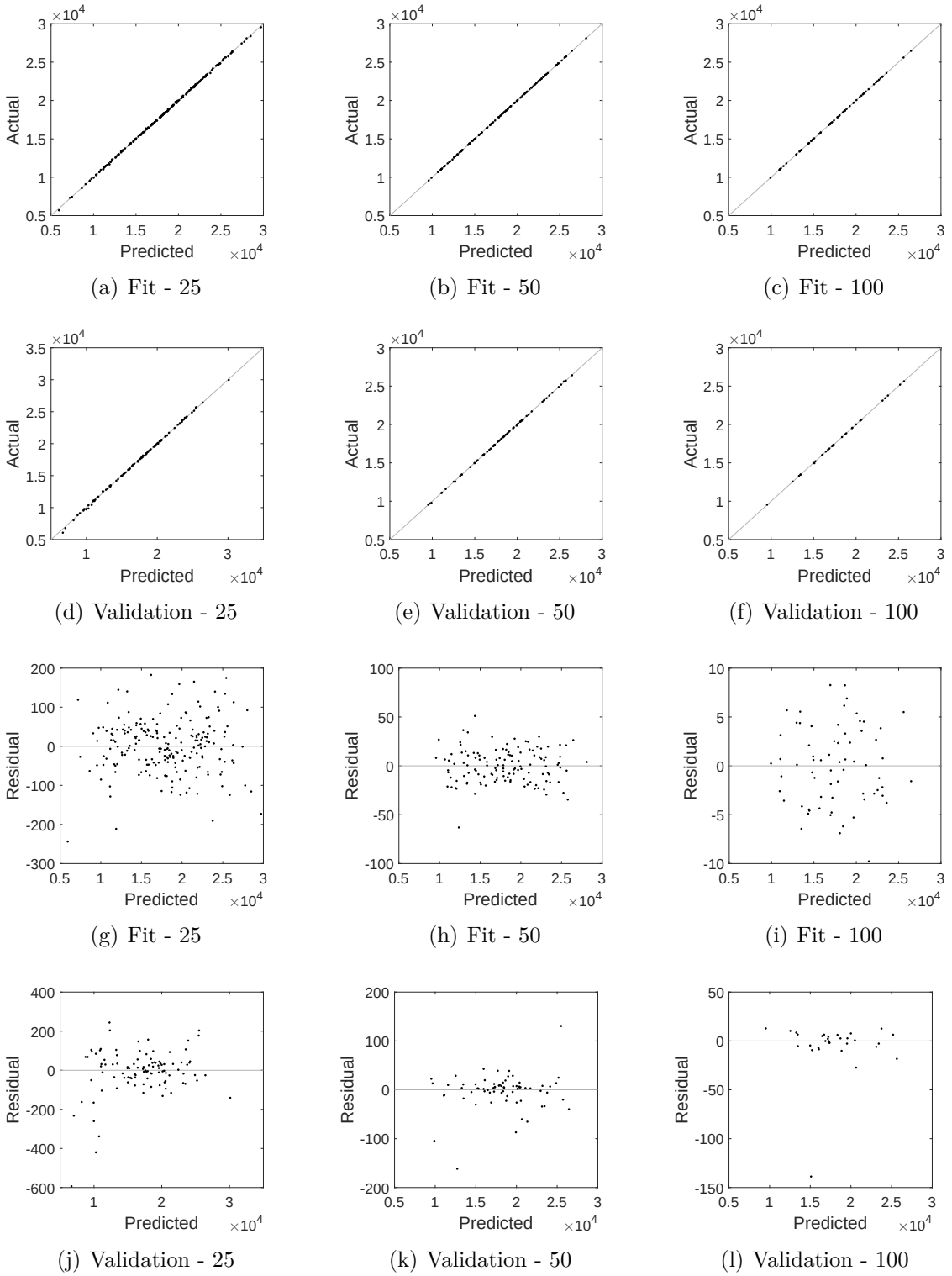
(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

Figure 116: Goodness of fit plots for polynomial regressions using 200 cases



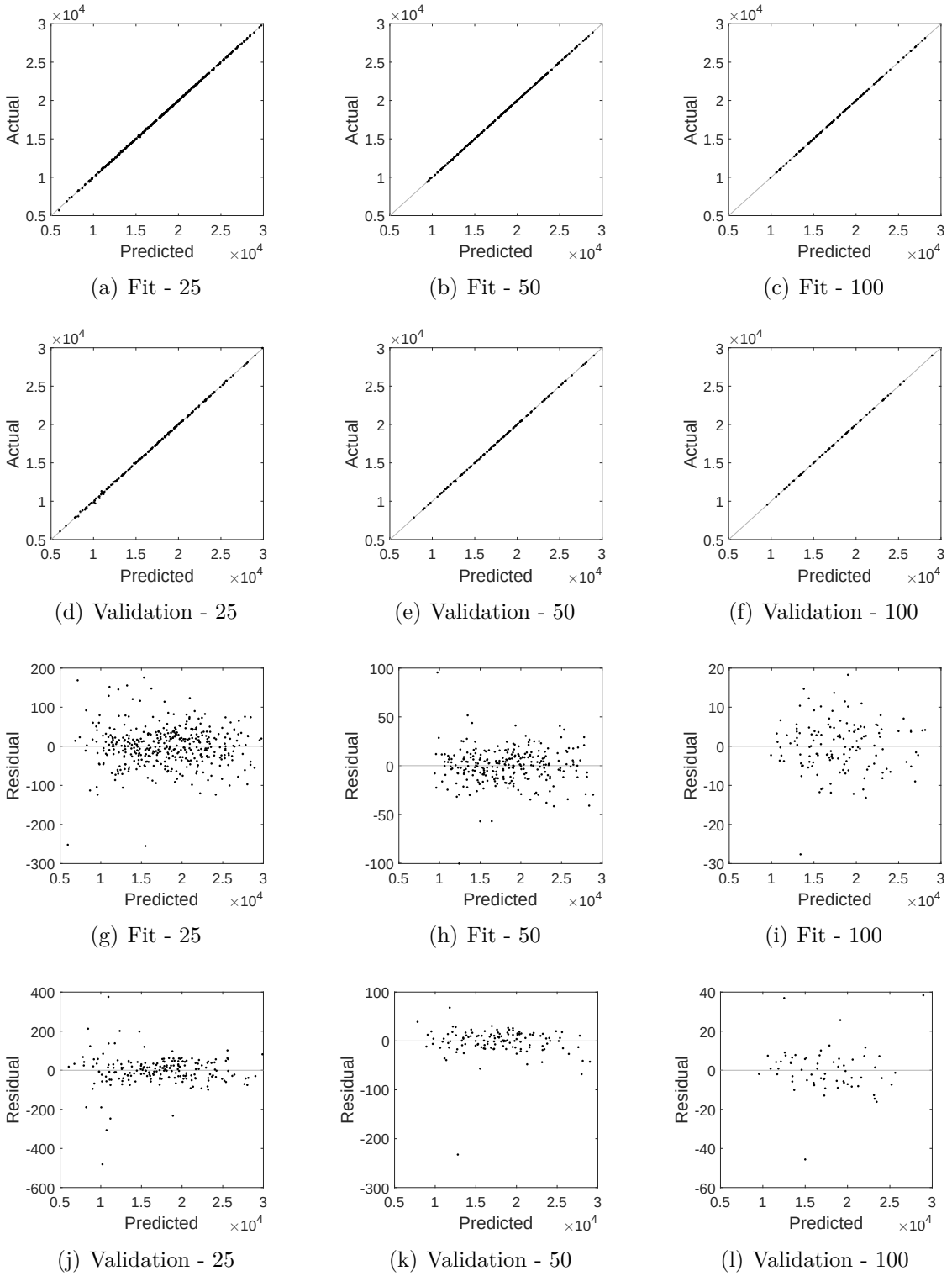
(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

Figure 117: Goodness of fit plots for polynomial regressions using 400 cases



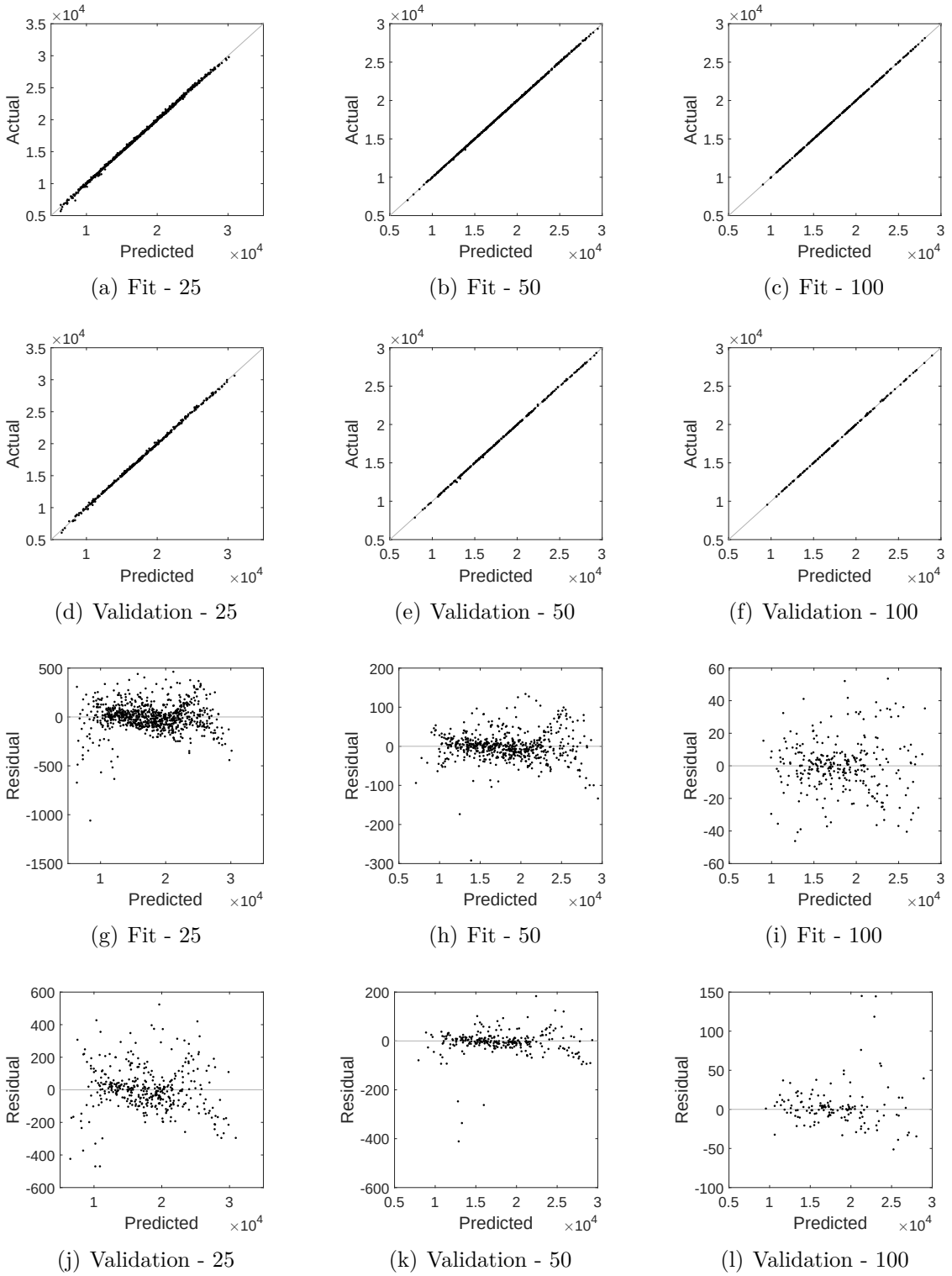
(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

Figure 118: Goodness of fit plots for polynomial regressions using 800 cases



(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

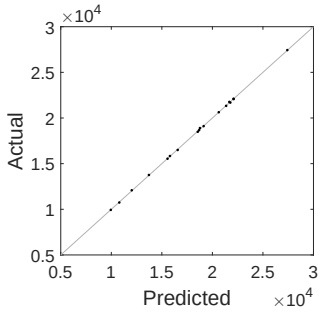
(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

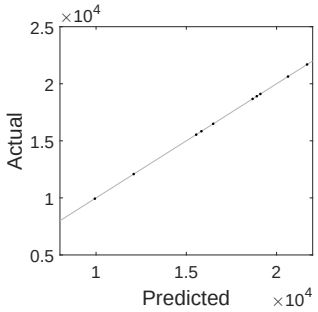
Figure 119: Goodness of fit plots for polynomial regressions using 1600 cases

D.4.3 Kriging

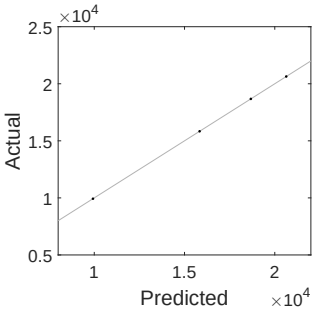
Figures 120 through 126 show the results for the kriging models. As in Section D.4.1, each figure represents a regression generated using an LHC design with the number of cases reported in the caption. The sub-figures (a)-(f) show the actual vs the predicted values. The sub-figures (g)-(l) show the residual by predicted. The labels on the sub-figures represent the number of repetitions required for each data point, 25, 50, or 100, and whether the figure is showing fit data or validation data. The validation plots represented with "n/a" signify that no validation points were available.



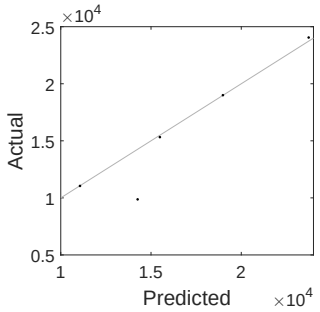
(a) Fit - 25



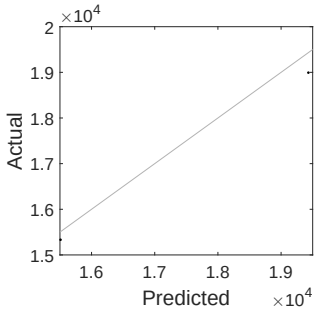
(b) Fit - 50



(c) Fit - 100



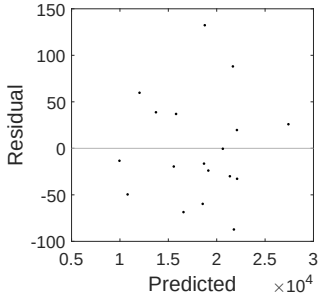
(d) Validation - 25



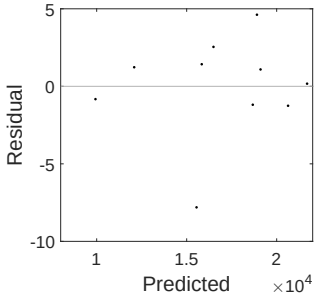
(e) Validation - 50

n/a

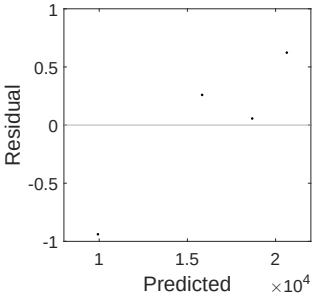
(f) Validation - 100



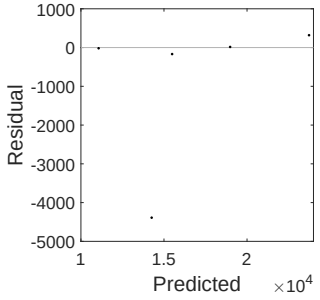
(g) Fit - 25



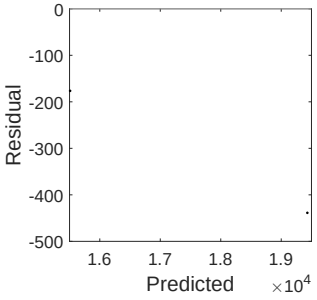
(h) Fit - 50



(i) Fit - 100



(j) Validation - 25

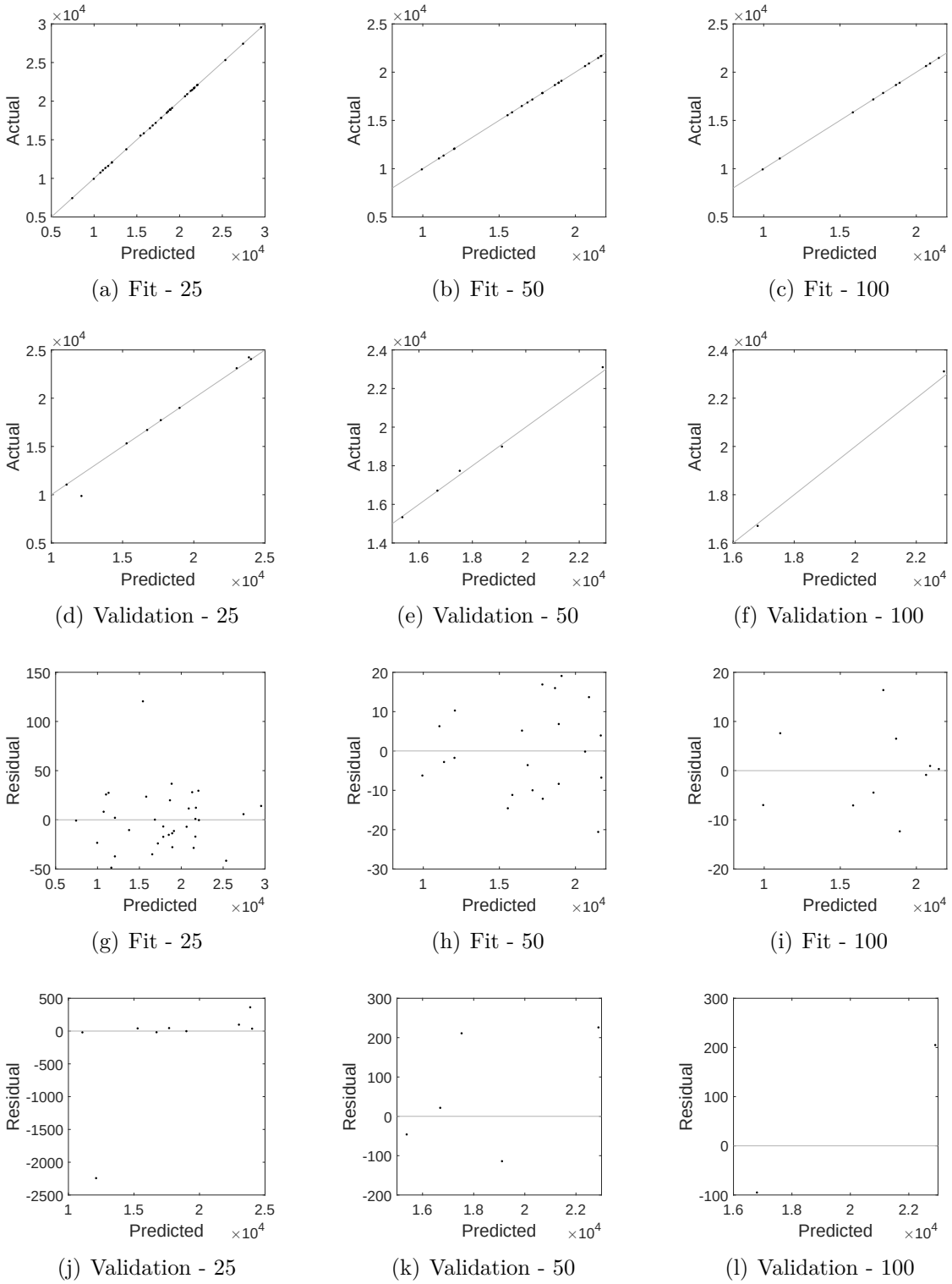


(k) Validation - 50

n/a

(l) Validation - 100

Figure 120: Goodness of fit plots for kriging models using 25 cases



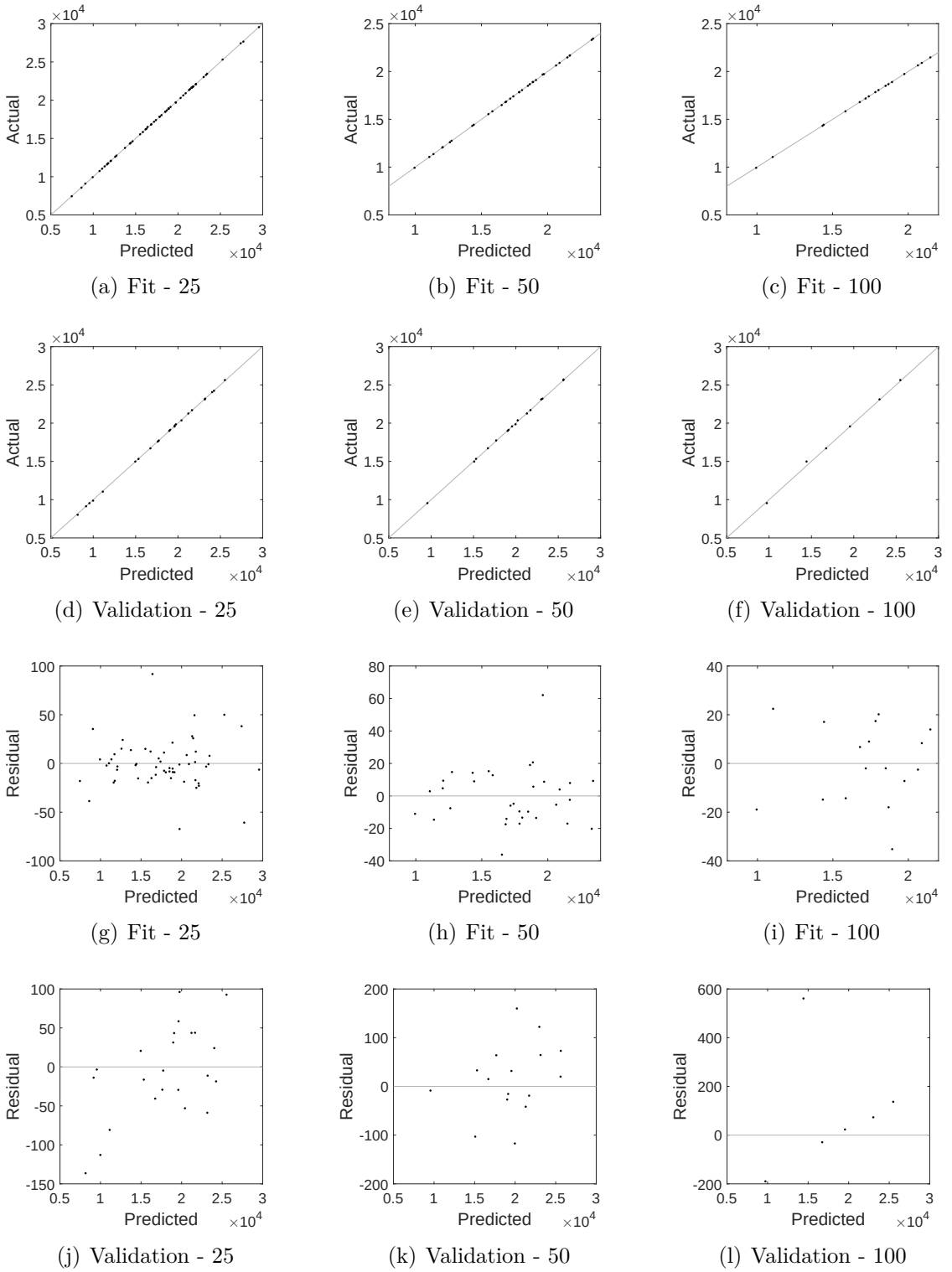
(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

Figure 121: Goodness of fit plots for kriging models using 50 cases



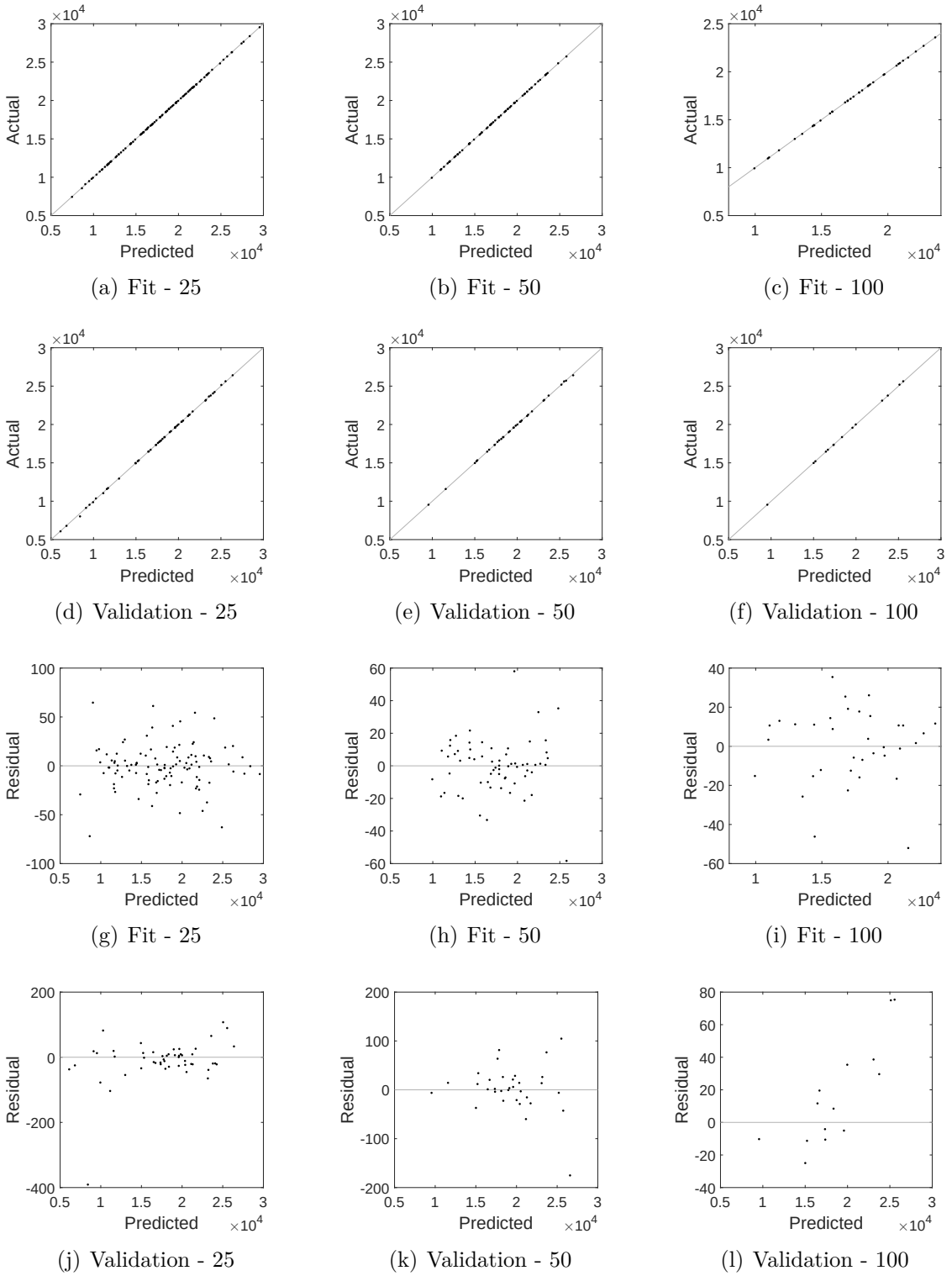
(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

Figure 122: Goodness of fit plots for kriging models using 100 cases



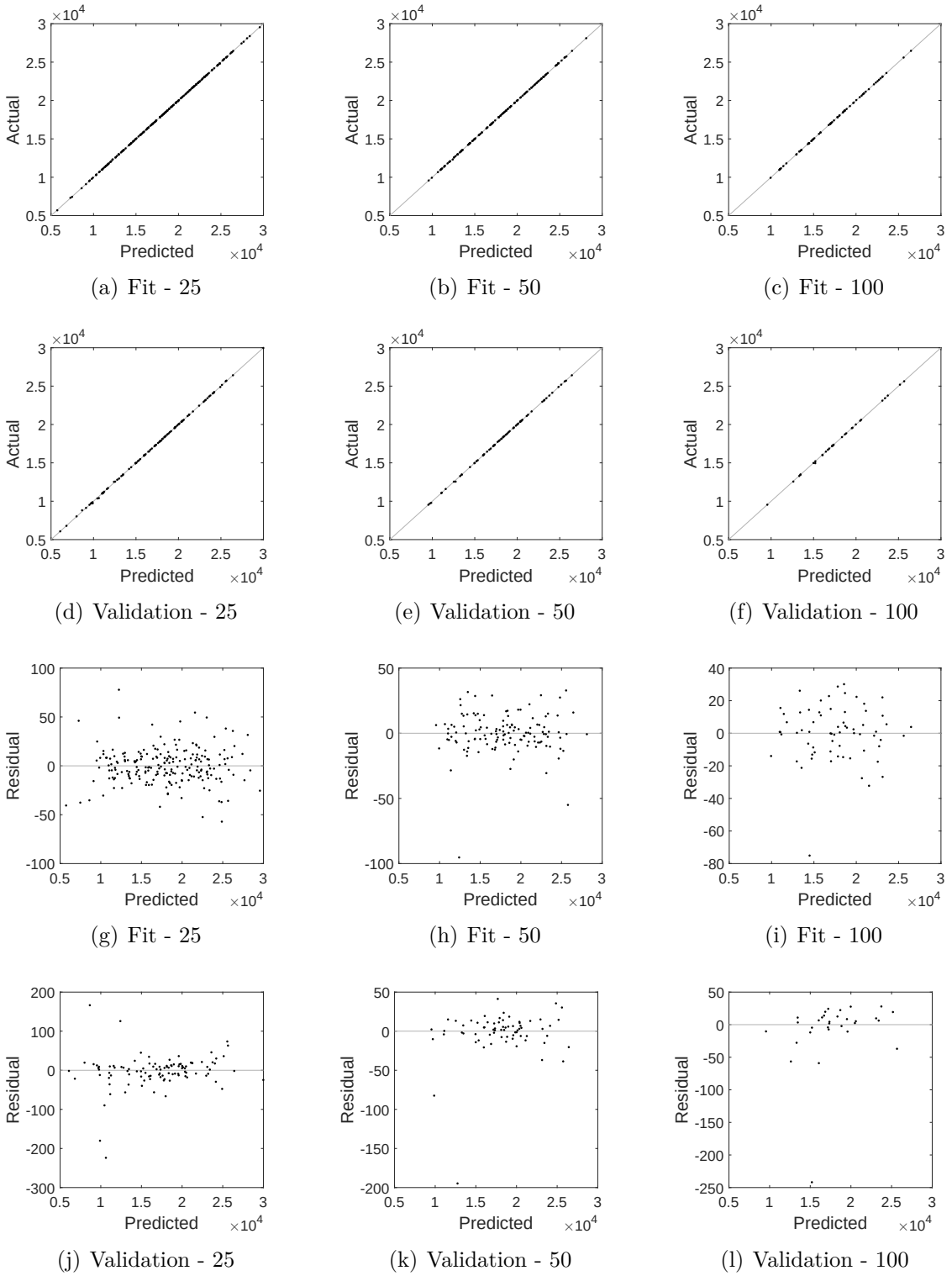
(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

Figure 123: Goodness of fit plots for kriging models using 200 cases



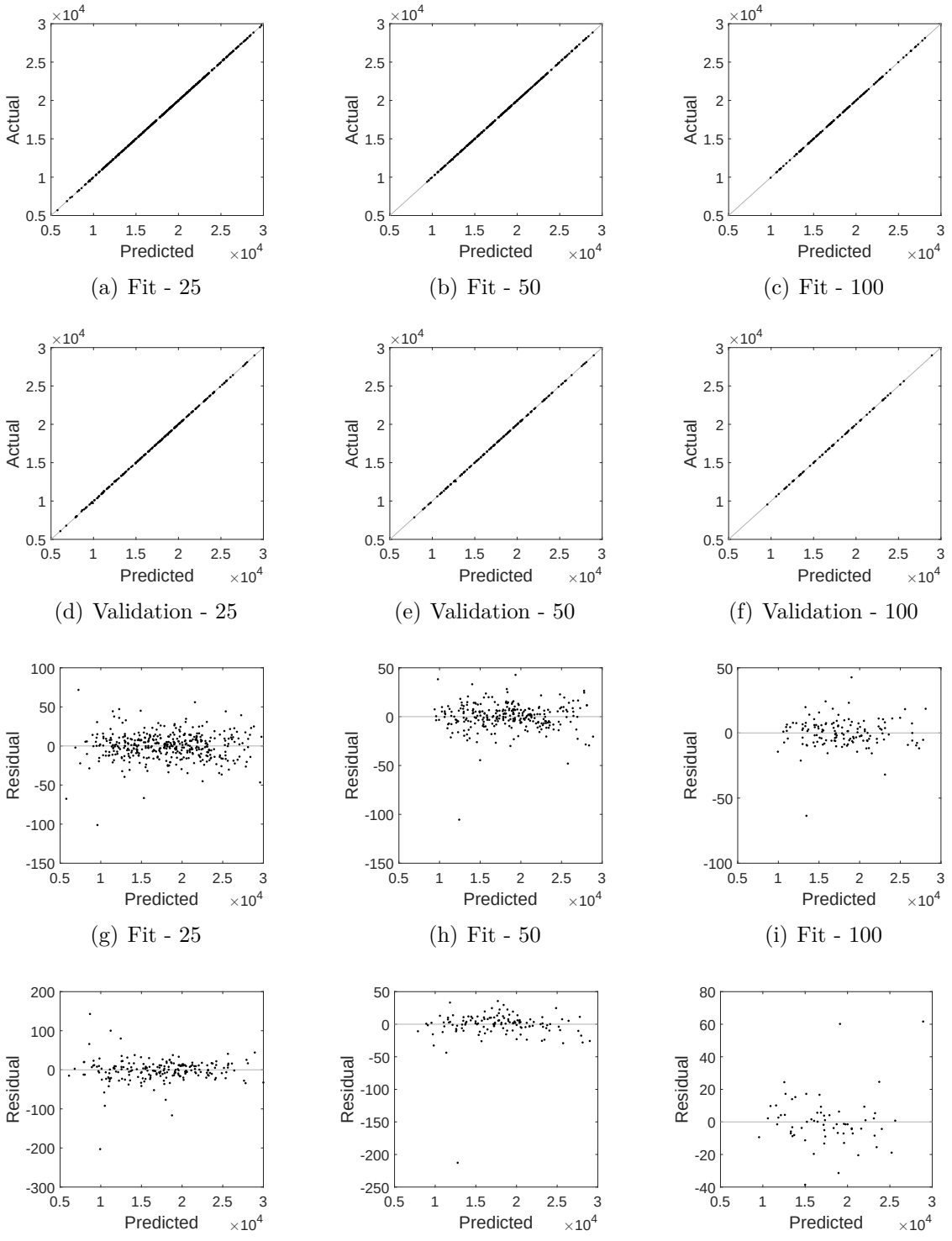
(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

Figure 124: Goodness of fit plots for kriging models using 400 cases



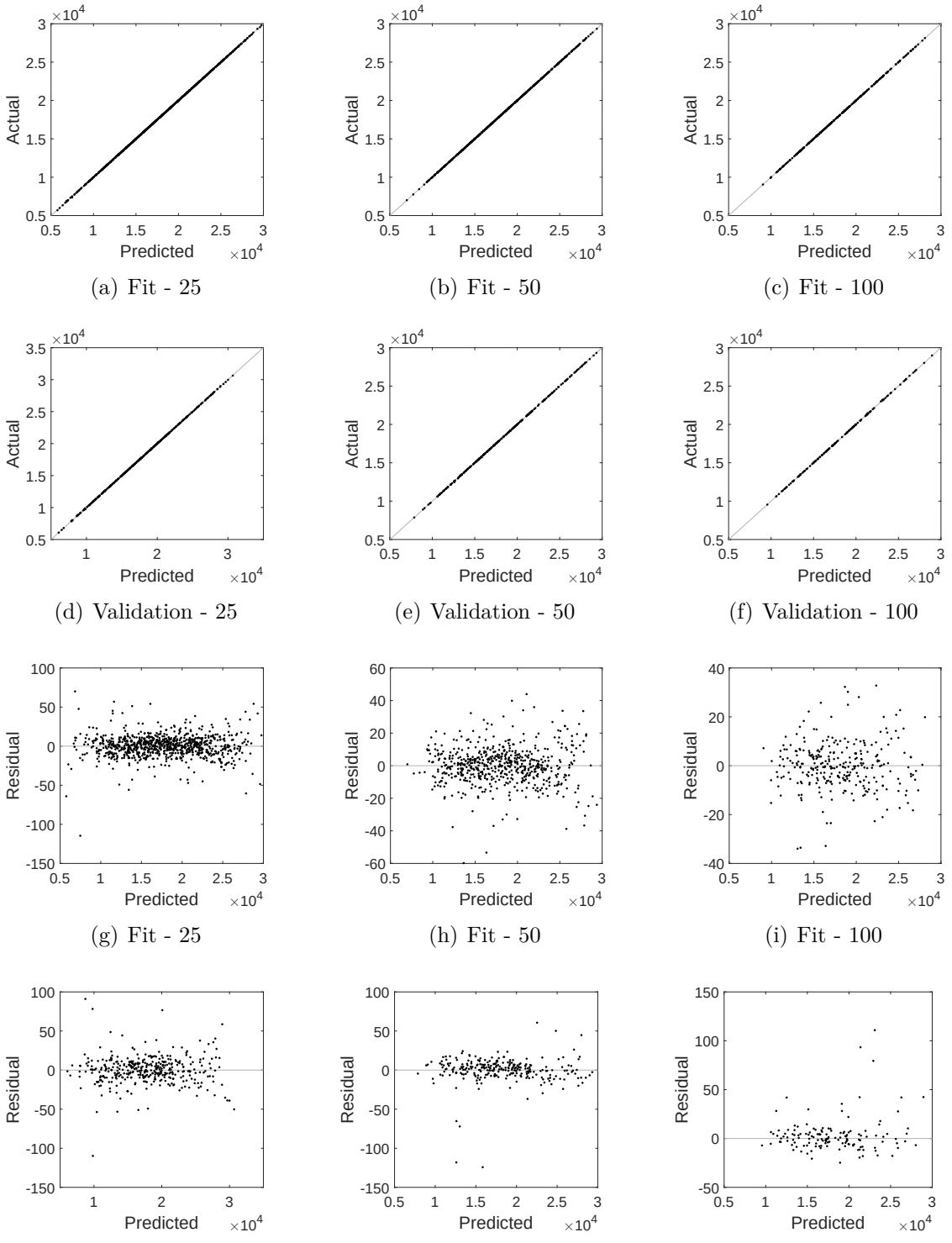
(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

Figure 125: Goodness of fit plots for kriging models using 800 cases



(a) Fit - 25 (b) Fit - 50 (c) Fit - 100

(d) Validation - 25 (e) Validation - 50 (f) Validation - 100

(g) Fit - 25 (h) Fit - 50 (i) Fit - 100

(j) Validation - 25 (k) Validation - 50 (l) Validation - 100

Figure 126: Goodness of fit plots for kriging models using 1600 cases

REFERENCES

[1] Addis, B., Cassioli, A., Locatelli, M., and Schoen, F., "A global optimization method for the design of space trajectories," *Computational Optimization and Applications*, vol. 48, no. 3, pp. 635–652, 2011.

[2] Akhtar, S. and Linshu, H., "Support vector machine based trajectory metamodel for conceptual design of multi-stage space launch vehicle," in *Computational Intelligence and Security*, pp. 528–535, Springer, 2005.

[3] Akhtar, S. and Linshu, H., "Support vector regression-driven multidisciplinary design optimization for multi-stage space launch vehicle considering throttling effect," in *44th AIAA Aerospace Sciences Meeting*, vol. 6, pp. 4089–4102, 2006.

[4] Anisi, D., "On-line trajectory planning using adaptive temporal discretization," tech. rep., Royal Institute of Technology, Sweden, 2006.

[5] Anisi, D. A., "Adaptive Node Distribution for On-line Trajectory Planning," 2006.

[6] Arora, J. S., *Introduction to Optimum Design*. Elsevier, Inc., second edition ed., 2004.

[7] Ascher, U. M. and Petzold, L. R., *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. Society for Industrial and Applied Mathematics, 1998.

[8] Atkinson, K. E., *An Introduction to Numerical Analysis*. John Wiley & Sons, Inc., 2nd ed., 1989.

[9] Balesdent, M., *Multidisciplinary Design Optimization of Launch Vehicles*. PhD thesis, École Centrale de Nantes, 2011.

[10] Balesdent, M., Bérend, N., Dépincé, P., and Chriette, A., "A survey of multidisciplinary design optimization methods in launch vehicle design," *Structural and Multidisciplinary optimization*, vol. 45, no. 5, pp. 619–642, 2012.

[11] Barton, R. R. and Meckesheimer, M., "Metamodel-based simulation optimization," *Handbooks in operations research and management science*, vol. 13, pp. 535–574, 2006.

[12] Bartz-Beielstein, T., Chiarandini, M., Paquete, L., and Preuss, M., *Experimental methods for the analysis of optimization algorithms*. Springer, 2010.

[13] Basak, D., Pal, S., and Patranabis, D. C., "Support vector regression," *Neural Information Processing-Letters and Reviews*, vol. 11, no. 10, pp. 203–224, 2007.

[14] Bate, R. R., Mueller, D. D., and White, J. E., *Fundamentals of Astrodynamics*. Dover Publications, Inc., 1st ed., 1971.

[15] Bayley, D. J., *DESIGN OPTIMIZATION OF SPACE LAUNCH VEHICLES USING A GENETIC ALGORITHM*. PhD thesis, Auburn University, 2007.

[16] Beirlant, J., Goegebeur, Y., Segers, J., and Teugels, J., *Statistics of extremes: theory and applications*. John Wiley & Sons, 2006.

[17] Ben-Asher, J. Z., *Optimal Control Theory with Aerospace Applications*. American institute of aeronautics and astronautics, 2010.

[18] Bertsekas, D. P., *Nonlinear programming*. Athena scientific, 1999.

[19] Betts, J. T., "Survey of numerical methods for trajectory optimization," *Journal of guidance, control, and dynamics*, vol. 21, no. 2, pp. 193–207, 1998.

[20] Betts, J. T., *Practical Methods for Optimal Control using Nonlinear Programming*. Siam, 2010.

[21] Betts, J. T. and Huffman, W. P., "Mesh refinement in direct transcription methods for optimal control," *Optimal Control Applications and Methods*, vol. 19, no. 1, pp. 1–21, 1998.

[22] Blair, J., Ryan, R., Schutzenhofer, L., and Humphries, W., "Launch vehicle design process: characterization, technical integration, and lessons learned," tech. rep., Marshall Space Flight Center, National Aeronautics and Space Administration, 2001.

[23] Blake, W., Rosema, C., Doyle, J., Auman, L., and Underwood, M., *Missile DATCOM*. Air Force Research Laboratory, March 2011.

[24] Boeing IV, D., "Payload planners guide," 1999.

[25] Bowman, A. W. and Azzalini, A., *Applied Smoothing Techniques for Data Analysis: The Kernel Approach with S-Plus Illustrations: The Kernel Approach with S-Plus Illustrations*. Oxford University Press, 1997.

[26] Box, G. E. and Wilson, K., "On the experimental attainment of optimum conditions," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 13, no. 1, pp. 1–45, 1951.

[27] Brauer, G. L., Cornick, D. E., Habeger, A., Petersen, F. M., and Stevenson, R., "Program to optimize simulate trajectories (post) volume i: Formulation manual," tech. rep., NASA Langley Research Center, 1975.

[28] Brauer, G. L., Cornick, D. E., and Stevenson, R., "Capabilities and applications of the program to optimize simulated trajectories (post)," tech. rep., Martin Marietta Corporation, 1977.

[29] Braun, R., Moore, A., and Kroo, I., "Use of the collaborative optimization architecture for launch vehicle design," in *Proceedings of the 6th AIAA/-NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 1996.

[30] Broomhead, D. S. and Lowe, D., "Radial basis functions, multi-variable functional interpolation and adaptive networks," tech. rep., DTIC Document, 1988.

[31] Bryson, A. E. J. and Ho, Y.-C., *Applied Optimal Control: Optimization, Estimation, and Control*. New York: Wiley, 1975.

[32] Bryson, A. E. and Ho, Y.-C., *Applied optimal control: optimization, estimation and control*. Blaisdell Publishing Company, 1975.

[33] Buhmann, M. D., "Radial basis functions," *Acta Numerica 2000*, vol. 9, pp. 1–38, 2000.

[34] Castellini, F., *Multidisciplinary Design Optimization for Expendable Launch Vehicles*. PhD thesis, Politecnico di Milano, 2012.

[35] Castellini, F., Lavagna, M. R., Riccardi, A., and Buskens, C., "Multidisciplinary design optimization models and algorithms for space launch vehicles," in *AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, Series Multidisciplinary Design Optimization Models and Algorithms for Space Launch Vehicles, (Fort Worth, TX), 2010.

[36] Castellini, F., Lavagna, M. R., Riccardi, A., and Büskens, C., "Quantitative assessment of multidisciplinary design models for expendable launch vehicles," *Journal of Spacecraft and Rockets*, vol. 51, no. 1, pp. 343–359, 2013.

[37] Cha, S.-H., "Comprehensive survey on distance/similarity measures between probability density functions," *City*, vol. 1, no. 2, p. 1, 2007.

[38] Charras-Garrido, M. and Lezaud, P., "Extreme value analysis: an introduction," *Journal de la Société Française de Statistique*, vol. 154, no. 2, pp. 66–97, 2013.

[39] Chenglong, H., Xin, C., and Leni, W., "Optimizing rlv ascent trajectory using pso algorithms," in *Systems and Control in Aerospace and Astronautics, ISSCAA 2008*, Series Optimizing RLV Ascent Trajectory Using PSO Algorithms, 2008.

[40] Ching, H., Ko, P.-H., Kong, C., and Du, R., "Adaptive node placement for optimal control." ICROS-SICE International Joint Conference 2009, August 2009.

[41] Clarke, F., *Functional analysis, calculus of variations and optimal control*, vol. 264. Springer Science & Business Media, 2013.

[42] Clarke, S. M., Griebsch, J. H., and Simpson, T. W., "Analysis of support vector regression for approximation of complex engineering analyses," *Journal of mechanical design*, vol. 127, no. 6, pp. 1077–1087, 2005.

[43] Combes, C. and Dussauchoy, A., "Generalized extreme value distribution for fitting opening/closing asset prices and returns in stock-exchange," *Operational Research*, vol. 6, no. 1, pp. 3–26, 2006.

[44] Contreras-Reyes, J. E. and Arellano-Valle, R. B., "Kullback–leibler divergence measure for multivariate skew-normal distributions," *Entropy*, vol. 14, no. 9, pp. 1606–1626, 2012.

[45] Conway, B. A., *Spacecraft Trajectory Optimization*, vol. 29. Cambridge University Press, 2010.

[46] Conway, B. A., "A survey of methods available for the numerical optimization of continuous dynamic systems," *Journal of Optimization Theory and Applications*, vol. 152, no. 2, pp. 271–306, 2012.

[47] Corman, J. A. and German, B. J., "A comparison of metamodeling techniques for engine cycle design data," in *AIAA/ISSMO Multidisciplinary Analysis Optimizaiton Conference*, (Fort Worth, Texas), September 2010.

[48] Creech, D. M., Threet Jr, G. E., Philips, A. D., Waters, E. D., and Baysinger, M., "Modular approach to launch vehicle design based on a common core element," in *AIAA SPACE 2010 Conference & Exposition*, Anaheim, California, USA, 2010.

[49] Csiszár, S., "Optimization algorithms (survey and analysis)," in *Logistics and Industrial Informatics, 2007. LINDI 2007. International Symposium on*, pp. 185–188, IEEE, 2007.

[50] Curtis, H., *Orbital mechanics for engineering students*. Butterworth-Heinemann, 2013.

[51] Darby, C. L. and Rao, A. V., "A mesh refinement algorithm for solving optimal control problems using pseudospectral methods," *Proceedings of the AIAA*, 2009.

[52] Das, S. and Suganthan, P. N., "Differential evolution: A survey of the state-of-the-art," *Evolutionary Computation, IEEE Transactions on*, vol. 15, no. 1, pp. 4–31, 2011.

[53] De Haan, L. and Ferreira, A., *Extreme value theory: an introduction*. Springer, 2007.

[54] Dixit, M., Upadhyay, N., and Silakari, S., "An exhaustive survey on nature inspired optimization algorithms," *International Journal of Software Engineering and Its Applications*, vol. 9, no. 4, pp. 91–104, 2014.

[55] Efron, B., "Bootstrap methods: another look at the jackknife," *The annals of Statistics*, vol. 7, pp. 1–26, 1979.

[56] Elnagar, G., Kazemi, M. A., and Razzaghi, M., "The pseudospectral legendre method for discretizing optimal control problems," *IEEE Transactions on Automatic Control*, vol. 40, no. 10, pp. 1793–1796, 1995.

[57] Engelund, W. C., Stanley, D. O., Lepsch, R. A., McMillin, M. M., and Unal, R., "Aerodynamic configuration design using response surface methodology analysis," *NASA STI/Recon Technical Report A*, vol. 94, p. 10718, 1993.

[58] Epix Analytics, *ModelAssist for Crystal Ball: Non-parametric and parametric distributions*, 2010. Page Reference Number: R-M0320-A.

[59] Frank, J., "Collocation methods," 2008.

[60] Futron Corporation, *Space Transportation Costs: Trends in Price Per Pound to Orbit 1990-2000*, 2002.

[61] Geeraert, J. L., "Sensitivity analysis of 2nd stage launch vehicle guidance algorithms," tech. rep., Georgia Institute of Technology, Aerospace Systems Design Lab, 2012.

[62] Geluk, J. L. and de Haan, L., "On bootstrap sample size in extreme value theory," *Publications de l'Institut Mathematique*, vol. 71, no. 85, pp. 21–26, 2002.

[63] Gerdts, M., "Direct shooting method for the numerical solution of higher-index dae optimal control problems," *Journal of Optimization Theory and Applications*, vol. 117, no. 2, pp. 267–294, 2003.

[64] Germain, B. S. and Feld, K., "Investigation of reusable booster system (rbs) trajectories," 2011.

[65] Ghosh, A., *Trajectory Optimization of Satellite Launch Vehicle using Evolutionary Algorithms*. PhD thesis, Jadavpur University Kolkata, India, 2014.

[66] Gilli, M. and Kellezi, E., "An application of extreme value theory for measuring financial risk," *Computational Economics*, vol. 27, no. 2-3, pp. 207–228, 2006.

[67] Giri, R. and Ghose, D., "Differential evolution based ascent phase trajectory optimization for a hypersonic vehicle," in *Swarm, Evolutionary, and Memetic Computing*, Chennai, India: First International Conference on Swarm, Evolutionary, and Memetic Computing, SEMCCO 2010, 2010.

[68] Gong, Q., Fahroo, F., and Ross, I. M., "Spectral algorithm for pseudospectral methods in optimal control," *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 3, pp. 460–471, 2008.

[69] Grant, M. J. and Braun, R. D., "Rapid indirect trajectory optimization for conceptual design of hypersonic missions," *Journal of Spacecraft and Rockets*, vol. 52, no. 1, pp. 177–182, 2014.

[70] Griffiths, D. and Higham, D. J., *Numerical Methods for Ordinary Differential Equations: Initial Value Problems*. Springer Science & Business Media, 2010.

[71] Gumbel, E., *Statistics of extremes, 1958*. Columbia Univ. press, New York, 1958.

[72] Hager, W. W. and Zhang, H., "A survey of nonlinear conjugate gradient methods," *Pacific journal of Optimization*, vol. 2, no. 1, pp. 35–58, 2006.

[73] Hargraves, C. R. and Paris, S. W., "Direct trajectory optimization using nonlinear programming and collocation," *Journal of Guidance, Control, and Dynamics*, vol. 10, no. 4, pp. 338–342, 1986.

[74] Hargraves, C. R., Paris, S. W., and Vlases, W. G., "Otis past, present and future," in *AIAA*, Series OTIS Past, Present and Future, 1993.

[75] Hart, W. E., *Adaptive global optimization with local search*. PhD thesis, University of California, San Diego, 1994.

[76] Hastie, T., , Tibshirani, R., and Friedman, J., *The elements of statistical learning*. Springer Sceicne and Business Media, 2009.

[77] Hodge, V. J. and Austin, J., "A survey of outlier detection methodologies," *Artificial Intelligence Review*, vol. 22, no. 2, pp. 85–126, 2004.

[78] Hull, D. G., "Conversion of optimal control problems into parameter optimization problems," *Journal of Guidance, Control, and Dynamics*, vol. 20, no. 1, pp. 57–60, 1997.

[79] Humphries Sr, W., Holland, W., and Bishop, R., "Information flow in the launch vehicle design/analysis process." National Aeronautics and Space Administration: Marshall Space Flight Center, 1999.

[80] Hüsler, J., Cruz, P., Hall, A., and Fonseca, C. M., "On optimization and extreme value theory," *Methodology and Computing in Applied Probability*, vol. 5, no. 2, pp. 183–195, 2003.

[81] Isakowitz, S. J., Hopkins, J. P., and Hopkins, J. B., *International reference guide to space launch systems*. American Institute of Aeronautics and Astronautics, 4th ed., 2004.

[82] Jain, S., *Multiresolution strategies for the numerical solution of optimal control problems*. PhD thesis, Georgia Institute of Technology, 2008.

[83] Jamil, M. and Yang, X.-S., "A literature survey of benchmark functions for global optimisation problems," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 4, no. 2, pp. 150–194, 2013.

[84] Jankauskas, L. and McLafferty, S., "Bestfit, distribution-fitting software by palisade corporation," in *Proceedings of the 28th conference on Winter simulation*, pp. 551–555, IEEE Computer Society, 1996.

[85] Jin, R., Chen, W., and Simpson, T. W., "Comparative studies of metamodelling techniques under multiple modelling criteria," *Structural and Multidisciplinary Optimization*, vol. 23, no. 1, pp. 1–13, 2001.

[86] Johnson, R. A., Miller, I., and Freund, J. E., *Probability and statistics for engineers*. Prentice-Hall, 2011.

[87] Joyce, J. M., "Kullback-leibler divergence," in *International Encyclopedia of Statistical Science*, pp. 720–722, Springer, 2011.

[88] Kamada, R., *Trajectory Optimization Strategies for Supercavitating Vehicles*. PhD thesis, Georgia Institute of Technology, 2005.

[89] Kaya, C. Y. and Maurer, H., "A numerical method for nonconvex multiobjective optimal control problems," *Computational Optimization and Applications*, vol. 57, no. 3, pp. 685–702, 2014.

[90] Kelley, C. T., *Iterative methods for optimization*. Siam, 1999.

[91] Kirk, D. E., *Optimal control theory: an introduction*. Courier Corporation, 2012.

[92] Ko, P.-H. and Du, R., "Adaptive node placement for optimal control on system of multi-degree freedom." SICE Annual Conference 2010, August 2010.

[93] Kröse, B. and van der Smagt, P., "An introduction to neural networks," 1996.

[94] Kullback, S. and Leibler, R. A., "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[95] Kyselỳ, J., "A cautionary note on the use of nonparametric bootstrap for estimating uncertainties in extreme-value models," *Journal of Applied Meteorology and Climatology*, vol. 47, no. 12, pp. 3236–3251, 2008.

[96] Lafleur, J. M., Fleming, E. S., and Saleh, J. H., "Response surface equations for expendable launch vehicle payload mass capability," *Journal of Spacecraft and Rockets*, vol. 49, no. 1, pp. 185–189, 2012.

[97] Laurikkala, J., Juhola, M., Kentala, E., Lavrac, N., Miksch, S., and Kavsek, B., "Informal identification of outliers in medical data," in *Fifth International Workshop on Intelligent Data Analysis in Medicine and Pharmacology*, pp. 20–24, Citeseer, 2000.

[98] Lee, J.-W., Jeon, K.-S., Byun, Y.-H., and Kim, S.-J., "Optimal space launcher design using a refined response surface method," in *Fuzzy Systems and Knowledge Discovery*, pp. 1081–1091, Springer, 2005.

[99] Li, J., Baoyin, H., and Vadali, S. R., "Autonomous rendezvous architecture design for lunar lander," *Journal of Spacecraft and Rockets*, 2015.

[100] Li, Y., Ng, S. H., Xie, M., and Goh, T., "A systematic comparison of metamodeling techniques for simulation optimization in decision support systems," *Applied Soft Computing*, vol. 10, no. 4, pp. 1257–1273, 2010.

[101] Liberti, L., "Introduction to global optimization," tech. rep., LIX, ÂtEcole Polytechnique, 2008.

[102] Liu, F., Hager, W. W., and Rao, A. V., "An hp mesh refinement method for optimal control using discontinuity detection and mesh size reduction," in *53rd IEEE Conference on Decision and Control*, pp. 5868–5873, IEEE, 2014.

[103] Liu, F., Hager, W. W., and Rao, A. V., "Adaptive mesh refinement method for optimal control using nonsmoothness detection and mesh size reduction," *Journal of the Franklin Institute*, vol. 352, no. 10, pp. 4081–4106, 2015.

[104] Longuski, J. M., Guzmán, J. J., and Prussing, J. E., *Optimal Control with Aerospace Applications*. Springer, 2014.

[105] Lurie, D., Abramson, L., and Vail, J., *Applying Statistics*. United States Nuclear Rgulatory Comission, 2011.

[106] Lyu, Z., Xu, Z., and Martins, J., "Benchmarking optimization algorithms for wing aerodynamic design optimization," in *8th International Conference on Computational Fluid Dynamics (ICCFD8)*, 2014.

[107] Marler, R. T. and Arora, J. S., "Survey of multi-objective optimization methods for engineering," *Structural and multidisciplinary optimization*, vol. 26, no. 6, pp. 369–395, 2004.

[108] Mathews, J. H. and Fink, K. D., *Numerical Methods Using Matlab*. Prentice-Hall, Inc., 3rd ed., 1999.

[109] The MathWorks, Inc., Natick, MA, United States, *MATLAB and the Optimization Toolbox Release 2014a.*

[110] Mavris, D., "Class lectures advanced design methods ae 6373," tech. rep., Georgia Institute of Technology, 2010.

[111] Mavris, D. N., Baker, A. P., and Schrage, D. P., "Ippd through robust design simulation for an affordable short haul civil tiltrotor," 1997.

[112] Mavris, D. N. and DeLaurentis, D. A., "Methodology for examining the simultaneous impact of requirements, vehicle characteristics, and technologies on military aircraft design," *22nd International Congress of Aeronautical Sciences*, 2000.

[113] Maydeu-Olivares, A. and Garcia-Forero, C., "Goodness-of-fit testing," *International Encyclopedia of Education*, vol. 7, no. 1, pp. 190–196, 2010.

[114] Murdoch, D. J., Tsai, Y.-L., and Adcock, J., "P-values are random variables," *The American Statistician*, vol. 62, no. 3, pp. 242–245, 2008.

[115] Murray, W., Saunders, M. A., and Gill, P. E., "User's guide for snopt version 7: Software for large-scale nonlinear programming," tech. rep., Systems Optimization Laboratory, Stanford University, 2008.

[116] Musk, E., "Senate testimony may 5, 2004," May 5, 2004 2004.

[117] Myers, R. H., Montgomery, D. C., and Anderson-Cook, C. M., *Response surface methodology: process and product optimization using designed experiments*, vol. 705. John Wiley & Sons, 2009.

[118] Nadarajah, S., "Fundamental of extreme value theory," tech. rep., University of Manchester, 2016.

[119] Nelson, D., "Qualitative and quantitative assesment of optimal tarjectories by implicit simulation (otis) and program to optimize simulated trajectories (post)," tech. rep., Georgia Institute of Technology, 2001.

[120] NIST/SEMANTECH, "*e-Handbook of Statistical Methods*," 2012. http://www.itl.nist.gov/div898/handbook/, October 9, 2014.

[121] Olds, J., "The suitability of selected multidisciplinary design and optimization techniques to conceptual aerospace vehicle design," in *AIAA*, pp. 92–4791, 1992.

[122] Olds, J., Bradford, J., Charania, A., Ledsinger, L., McCormick, D., and Sorensen, K., "Hyperion: an ssto vision vehicle concept utilizing rocket-based combined cycle propulsion," *9th International space Plances and Hypersonic Systems and Technologies Conference*, pp. 99–4944, 1999.

[123] Olds, J. R. and Walberg, G., "Multidisciplinary design of a rocket-based combined-cycle ssto launch vehicle using taguchi methods," in *AIAA/AHS/ASEE Aerospace Design Conference*, (Irvine CA), pp. 93–1096, 1993.

[124] Paterlini, S. and Krink, T., "Differential evolution and partical swarm optimisation in partitional clustering," *Computational Statistics and Data Analysis*, vol. 50, no. 5, pp. 1220–1247, 2006.

[125] Perkins, F. M., "Derivation of linear-tangent steering laws." DTIC Document, 1966.

[126] Peter, J., Carrier, G., Bailly, D., Klotz, P., Marcelet, M., and Renac, F., "Local and global search methods for design in aeronautics," 2011.

[127] Pintér, J. D., *Global optimization: scientific and engineering case studies*, vol. 85. Springer, 2006.

[128] Powell, R. W., Striepe, S. A., Desai, P. N., Braun, R. D., Brauer, G. L., Cornick, D. E., Olson, D. W., and Petersen, F. M., "Program to optimize simulate trajectories (post) volume ii: Utilization manual," tech. rep., NASA Langley Research Center, 1997.

[129] Qazi, M.-U.-D., Linshu, H., and Permoon, M., "Hammersley sampling and support-vector-regression-driven launch vehicle design," *Journal of Spacecraft and Rockets*, vol. 44, no. 5, pp. 1094–1106, 2007.

[130] Qi, Y., "Bootstrap and empirical likelihood methods in extremes," *Extremes*, vol. 11, no. 1, pp. 81–97, 2008.

[131] Qian, P. Z., "Nested latin hypercube designs," *Biometrika*, p. asp045, 2009.

[132] Rafique, A. F., LinShu, H., Kamran, A., and Zeeshan, Q., "Multidisciplinary design of air launched satellite launch vehicle: Performance comparison of heuristic optimization methods," *Acta Astronautica*, vol. 67, no. 7, pp. 826–844, 2010.

[133] Rainey, L. B., *Space modeling and simulation: roles and applications throughout the system life cycle*. AIAA, 2004.

[134] Rao, A. V., "A survey of numerical methods for optimal control," *Advances in the Astronautical Sciences, pre-print*, vol. 135, no. 1, pp. 497–528, 2009.

[135] Rasky, D. J., Pittman, R. B., and Newfield, M. E., "The reusable launch vehicle challenge," in *AIAA Space 2006*, Series The Reusable Launch Vehicle Challenge, (San Jose, CA), 2006.

[136] Raymond H. Myers, Douglas C. Montgomery, C. M. A.-C., *Response Surface Methodology*. Wiley, 2009.

[137] Rea, J., "Launch vehicle trajectory optimization using a legendre pseudospectral method," in *Proceedings of the AIAA Guidance, Navigation and Control Conference, Paper No. AIAA*, vol. 5640, 2003.

[138] Riccardi, A., Castellini, F., Buskens, C., and Lavagna, M., "Global and local optimization approaches for launch vehicles ascent trajectory design," in *62nd International Astronautical Congress*, Series Global and Local Optimization Approaches for Launch Vehicles Ascent Trajectory Design, (Cape Town, SA), International Astronautical Federation, 2010.

[139] Riehl, J. P., Paris, S. W., and Sjauw, W. K., "Comparison of emplicit entegration methods for solving aerospace trajectory optimization problems," in *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, Series Comparison of Implicit Integration Methods for Solving Aerospace Trajectory Optimization Problems, (Keystone, CO), 2006.

[140] Ross, I. M. and Fahroo, F., "A perspective on methods for trajectory optimization," in *Proceedings of the AIAA/AAS Astrodynamics Specialist Conference, Monterey, CA*, pp. 5–8, 2002.

[141] Rowell, L. F., Braun, R. D., Olds, J. R., and Unal, R., "Multidisciplinary conceptual design optimization of space transportation systems," *Journal of Aircraft*, vol. 36, no. 1, pp. 218–226, 1999.

[142] Rowell, L. F. and Korte, J. J., "Launch vehicle design and optimization methods and priority for the advanced engineering environment." National Aeronautics and Space Administration: Langley Research Center, October 2003.

[143] Sahoo, P., *Probability and Mathematical Statistics*. University of Louisville, 2013.

[144] Salicrú, M., Morales, D., Menéndez, M., and Pardo, L., "On the applications of divergence type measures in testing statistical hypotheses," *Journal of Multivariate Analysis*, vol. 51, no. 2, pp. 372–391, 1994.

[145] Seywald, H., "Trajectory optimization based on differential inclusion (revised)," *Journal of Guidance, Control, and Dynamics*, vol. 17, no. 3, pp. 480–487, 1994.

[146] Shalizi, C. R., "Advanced data analysis from an elementary point of view," 2013.

[147] Sheppard, M., "Fit all valid parametric probability distributions to data," tech. rep., MATLAB File Exchange, 2012. Retrieved on 9/30/2014 from http://www.mathworks.com/matlabcentral/fileexchange/34943-fit-all-valid-parametric-probability-distributions-to-data/content/allfitdist.m.

[148] Shippey, B. M., *Trajectory Optimization using Collocation and Evolutionary Programming for Constrainted Nonlinear Dynamical Systems*. PhD thesis, University of Texas at Arlington, 2008.

[149] Simpson, T. W., Mauery, T. M., Korte, J. J., and Mistree, F., "Kriging models for global approximation in simulation-based multidisciplinary design optimization," *AIAA journal*, vol. 39, no. 12, pp. 2233–2241, 2001.

[150] Simpson, T. W., Poplinski, J., Koch, P. N., and Allen, J. K., "Metamodels for computer-based engineering design: survey and recommendations," *Engineering with computers*, vol. 17, no. 2, pp. 129–150, 2001.

[151] Singh, K. and Xie, M., "Bootstrap: a statistical method," *Unpublished Working Paper. Rutgers University,¡ http://www.stat.rutgers.edu/home/mxie/RCPapers/bootstrap.pdf*, 2008.

[152] Smith, R. L., "Statistics of extremes, with applications in environment, insurance and finance," *Extreme values in finance, telecommunications and the environment*, pp. 1–78, 2003.

[153] Smith, T. K., "Interim access to the international space station," Master's thesis, Utah State University, 2010.

[154] Smola, A. J. and Schölkopf, B., "A tutorial on support vector regression," *Statistics and computing*, vol. 14, no. 3, pp. 199–222, 2004.

[155] Snyman, J., *Practical mathematical optimization: an introduction to basic optimization theory and classical and new gradient-based algorithms*. Springer Science & Business Media, 2005.

[156] Sobieszczanski-Sobieski, J. and Haftka, R. T., "Multidisciplinary aerospace design optimization: survey of recent developments," *Structural optimization*, vol. 14, no. 1, pp. 1–23, 1997.

[157] Song, J. and Su, H., "The ascent trajectory optimization of two-stage-to-orbit aerospace plane based on pseudospectral method," *Procedia Engineering*, vol. 99, pp. 1044–1048, 2015.

[158] Spurlock, O. and Williams, C., "Duksup: A computer program for high thrust launch vehicle trajectory design & optimization," in *AIAA Propulsion and Energy Forum and Exposition*, 2014.

[159] Stanley, D. O., Engelund, W. C., Lepsch, R. A., McMillin, M., Wurster, K. E., Powell, R. W., Guinta, T., and Unal, R., "Rocket-powered single-stage vehicle configuration selection and design," *Journal of spacecraft and rockets*, vol. 31, no. 5, pp. 792–798, 1994.

[160] Steffens, M., Edwards, S., Diaz, M., Mavris, D. N., and Dees, P., "A method for launch vehicle performance analysis via surrogate modeling," in *AIAA Modeling and Simulation Technologies Conference*, p. 0677, 2016.

[161] Steffens, M. and Mavris, D., "Launch vehicle performance analysis using extreme value theory," in *Proceedings of the AIAA SPACE Conference and Exposition*, 2015.

[162] Steffens, M. J., "A combined global and local methodology for launch vehicle trajectory design-space exploration and optimization," Master's thesis, Georgia Institute of Technology, 2014.

[163] Stevenson, M. D., Hartong, A. R., Grandhi, R. V., Zweber, J. V., and Bhungalia, A. A., "Collaborative design environment for space launch vehicle design and optimization," tech. rep., DTIC Document, 2003.

[164] Stryk, O. v. and Burlisch, R., "Direct and indirect methods for trajectory optimization," *Annals of Operations Research*, vol. 37, no. 1, pp. 357–373, 1992.

[165] Suli, E., "Numerical solution of ordinary differential equations," April 2013. Lecture Notes at University of Oxford; Retrieved on 8/6/2014 from http://people.maths.ox.ac.uk/suli/nsodes.pdf.

[166] Torn, A. and Zilinskas, A., *Global Optimization*, vol. 350. Springer-Verlag, 1987.

[167] Turner, M. J., *Rocket and spacecraft propulsion: principles, practice and new developments*. Springer Science & Business Media, 2008.

[168] Ullah, R., Zhou, D.-Q., Zhou, P., Hussain, M., and Sohail, M. A., "An approach for space launch vehicle conceptual design and multi-attribute evaluation," *Aerospace science and technology*, vol. 25, no. 1, pp. 65–74, 2013.

[169] United Launch Alliance, *Delta IV Launch Services User's Guide*, February 2013.

[170] Vanderplaats, G. N., *Multidiscipline Design Optimization*. Colorado Springs, CO: Vanderplaats Research & Development, Inc., 2007.

[171] Vukelich, S. R., Stoy, S. L., Burns, K. A., Castillo, J. A., and Moore, M. E., *Missile DATCOM*. McDonnell Douglass Missile Systems Company, December 1988.

[172] Wang, L., Beeson, D., Wiggs, G., and Rayasam, M., "A comparison of metamodeling methods using practical industry requirements," in *The 47th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics, and materials conference, Newport, Rhode Island, USA*, pp. 1–4, 2006.

[173] Waschl, H., Kolmanovsky, I., Steinbuch, M., and Del Re, L., *Optimization and Optimal Control in Automotive Systems*, vol. 455. Springer, 2014.

[174] Wasserman, L., *All of nonparametric statistics*. Springer, 2006.

[175] Wasserstein, R. L. and Lazar, N. A., "The asa's statement on p-values: context, process, and purpose," *The American Statistician*, 2016. accepted preprint.

[176] Waters, E. D., Garcia, J., Beers, B., Philips, A., Holt, J. B., and Threet Jr, G. E., "Nasa advanced concepts office, earth-to-orbit team design process and tools," in *AIAA SPACE 2013 Conference and Exposition*, 2013.

[177] Wilhite, A., "Class lectures in spacecraft desing I AE 6322," 2011. Georgia Institute of Technology.

[178] Womersley, R., "Local and global optimization: Formulation, methods and applications," 2008.

[179] Young, D. A., Krevor, Z. C., Tanner, C., Thompson, R. W., and Wilhite, A. W., "Crew launch vehicle (clv) independent performance evaluation," *Space Systems Engineering Conference*, 2005.

[180] Zhao, Y. and Tsiotras, P., "Density functions for mesh refinement in numerical optimal control," *Journal of guidance, control, and dynamics*, vol. 34, no. 1, pp. 271–277, 2011.